



(19)

Europäisches Patentamt

European Patent Office

Office européen des brevets



(11)

EP 1 012 721 B1

(12)

## EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention  
of the grant of the patent:

11.07.2001 Bulletin 2001/28

(51) Int Cl.7: G06F 11/14

(21) Application number: 98915554.4

(86) International application number:  
PCT/US98/07458

(22) Date of filing: 15.04.1998

(87) International publication number:  
WO 99/13403 (18.03.1999 Gazette 1999/11)

### (54) DATA FILE STORAGE MANAGEMENT SYSTEM FOR SNAPSHOT COPY OPERATIONS

DATEISPEICHERVERWALTUNGSSYSTEM FÜR SCHNAPPSCHUSSKOPIEROPERATIONEN

SYSTEME DE GESTION DE MEMOIRE DE FICHIERS DE DONNEES POUR OPERATIONS DE  
COPIE SELECTIVE

(84) Designated Contracting States:  
DE FR

• MILILLO, Michael, Steven  
Louisville, CO 80027 (US)  
• O'BRIEN, John, Timothy  
Louisville, CO 80027 (US)

(30) Priority: 09.09.1997 US 925787

(74) Representative: Goodanew, Martin Eric et al  
MATHISEN, MACARA & CO.  
The Coach House  
6-8 Swakeleys Road  
Ickenham Uxbridge UB10 8BZ (GB)

(43) Date of publication of application:  
28.06.2000 Bulletin 2000/26

(73) Proprietor: STORAGE TECHNOLOGY  
CORPORATION  
Louisville, CO 80028-4309 (US)

(56) References cited:  
EP-A- 0 361 683 WO-A-96/12232

(72) Inventors:  
• BELSAN, Jay, Stuart  
Nederland, CO 80466 (US)

**Description****Field of the Invention**

**[0001]** This invention relates to a data storage subsystem which uses a snapshot copy process to copy a data file by duplicating the data file pointer in a mapping table to reference the original data file, such that the name used to identify the original data file and the name used to identify the copied data file are both mapped to the same physical data storage location. The integrity of the mapping table is maintained by the use of a redundantly stored two level mapping table which enables a data file to be relocated with only a simple change to a single entry in the mapping table, even when there are multiple data file names which refer to the data file.

**Problem**

**[0002]** It is a problem in computer systems and data storage subsystems to perform the data file copy operation in a manner that minimizes the use of processing resources and data storage space in memory. In the past, data files were copied in their entirety by the processor, such that two exact copies of the selected data file were resident in memory. This operation consumed twice the amount of memory for the storage of two identical copies of the data file and also required the intervention of the processor to effect the copy of the original data file.

**[0003]** An improvement over this copy process was the data file snapshot copy process disclosed in U.S. Patent No. 5,410,667, wherein a dynamically mapped virtual data storage subsystem stored data files received from a processor in backend data storage devices by mapping the processor assigned data file identifier to a logical address that identifies the physical storage location of the data. This dynamically mapped virtual data storage subsystem performed a copy of a data file by simply creating a duplicate data file pointer in a data file identifier in a mapping table to reference the original data file. In this dynamically mapped virtual data storage subsystem, the data files are referred to as virtual tracks and each data file is identified by a unique Virtual Track Address. The use of a mapping table provides the opportunity to replace the process of copying the entirety of a data file in the data storage devices with a process that manipulates the contents of the mapping table. A data file appears to have been copied if the name used to identify the original data file and the name used to identify the copy data file are both mapped to the same physical data storage location. This enables the processor to access the data file via two virtual track addresses while only a single physical copy of the data file resides on the backend data storage devices in the data storage subsystem. This process minimizes the time required to execute the copy operation and the amount of memory used since the copy operation is carried out by

creating a new pointer to the original data file and does not require any copying of the data file itself.

**[0004]** A problem with this copy system is that only the information stored in the mapping table identifies the

5 fact that the data file has been copied. If the mapping table is lost or its contents corrupted, the effect of the data file copy operation is lost. This problem can be avoided if the data storage subsystem stores the data file copy information in the data file itself to thereby replicate the mapping table data. However, such a process requires that the original data file be updated to reflect each copy operation that is executed and this overhead reduces the benefit of the pointer manipulation. Furthermore, any updates or changes to a data file in a log structured file system results in the entirety of the data file being rewritten in memory. Therefore, a log structured file system does not benefit from this replication of the mapping data.

**[0005]** A further problem with this copy system is that 20 the process of updating mapping table pointers to emulate the copying of data files requires a finite amount of processing time. It is therefore desirable for the processor to execute read and write operations during the process of updating the mapping table pointers, and this interleaved copy operation is termed an incremental copy process. This interleaving of operations can create instances in which the original data file (source area) is being written or the copy data file (target area) is being read or written before the mapping table update is completed. In this instance, data written to the original data file before the completion of the mapping table updates may or may not be reflected in the copy data file. The copy data file also now occupies its own unique physical space in memory (target area) since it does not correspond to the original data file. Similarly, data read from the copy data file before the completion of the mapping table updates may be the data file being copied or the data file which was stored in the target area before the copy operation was initiated. Furthermore, data written 30 to the copy data file before the completion of the mapping table updates may or may not be overwritten by the data file being copied. This results in uncertainty with regard to the correspondence between the original data file, its modified version and the copy data file. This incremental copy process is therefore less desirable than 35 a point in time copy process where the identity between the original and copy data files is ensured. With a point in time copy process, all data written to the copy source area before the initiation of the copy operation will appear in the copy target area, all data written to the copy target area before the initiation of the copy will be overwritten by data from the copy source area, no data written to the copy source area after the initiation of the copy will appear in the copy target area and no data written 40 to the copy target area after the initiation of the copy process will be overwritten by data from the copy source area. It is therefore desirable to provide an incremental copy process which interleaves read and write opera- 45 50 55

tions with the mapping table updates, but also preserves the point in time snapshot copy semantics to ensure correspondence of the copy and original data files during the mapping table updates.

[0006] Another problem with the incremental copy process is that this process can bypass security policies that block unauthorized data access. For example, a certain group of users may have access to a data file that was previously stored in the memory location now used for the copy data file. These users can access the copy data file that is being written to this area as part of the incremental copy process if the mapping table pointers, with its file access permissions, has not yet been updated. Thus, the access permission process is not synchronized with the copy process and users can access files for which they are not authorized.

[0007] Thus, there presently is no snapshot copy process available that both efficiently ensures the reliability of the mapping table data and also performs an incremental copy process which preserves the point in time copy semantics to ensure copy data file correspondence to the original data file.

### Solution

[0008] The above described problems are solved and a technical advance achieved by the present data file storage management system for snapshot copy operations. This system maintains a two level mapping table which enables the data files to be copied using the snapshot copy process but only requires the update of a single corresponding mapping table entry when the physical location of the data file instance is changed. The snapshot copy updates to the contents of the first level of the two level mapping table are stored on the backend data storage devices to provide a record of the snapshot copy operation which can be used to recover the correct contents of the mapping table. This record of the snapshot copy operation remains valid even though the physical location of a copied data file instance is subsequently changed. Furthermore, the physical storage space holding the updated portions of the first level of the two level mapping table can be managed using techniques like those used to manage the physical storage space holding data file instances. In addition, the synchronization of the snapshot copy operation with the reading and writing of data to the original and copy data files is maintained by detecting accesses to the original data file or the copy file during the time that the snapshot copy process is being executed and the mapping table is being updated. Mapping table updates resulting from the snapshot copy operation are delayed until all mapping table updates resulting from earlier data file write operations have been completed and any attempt to update the mapping table to reflect data written to the original data file or the copy data file that occurs after initiation of the snapshot copy operation must wait until the first set of mapping table pointers have been updated.

[0009] The present data file storage management system for snapshot copy operations is implemented in a dynamically mapped virtual data storage subsystem to maintain data file copy integrity in snapshot copy operations.

5 The data storage subsystem is connected to at least one processor and functions to store data files for the processor in the backend data storage devices that are part of the data storage subsystem. The processor assigns a virtual track address to a data file which is transmitted to the data storage subsystem for storage in an allocated physical storage location in the backend data storage devices. The assignment of a physical storage location on the backend data storage devices is effected by a controller contained within the data storage subsystem, which defines the correspondence between the processor assigned virtual track address and the logical address of the stored data file. A mapping table translates the virtual track address into a logical address which identifies the location of the data file on a physical disk drive. The location of the data file changes as the data storage subsystem free space collection process moves the data file to create free space into which new data can be written. It is therefore insufficient to store the translation from a virtual address to a logical address as a means of preserving a record of the mapping table updates, since the free space collection process changes the physical location of the data file but does not update these translations. The data files stored on the disk drives must therefore contain information that is independent of the logical address at which the data file presently being copied is stored. This enables the disk stored information to remain valid even though the physical location of the data file may change over time. This is accomplished by the use of a two level mapping architecture. The first level of mapping tables maps a virtual address to an immutable name which identifies a unit of data, such as a virtual track. The second level of mapping maps the immutable name to a given logical address. The snapshot copy operation operates on the first level of the mapping table to create multiple copies of the virtual track, thereby eliminating the need to associate with each virtual track address a mapping table entry which contains a logical address.

[0010] In a snapshot copy operation, to provide the illusion of there being two independent copies of the data, any write to one of the copies must leave the other copy unchanged. In a log structured file system, the writing of data or changes to a data file results in the data file being written to a different location in memory. If the original data file is accessible by a plurality of virtual addresses, it must remain in its original form for access by the remaining users, while the changes to this data file must be reflected in a new copy of the data file which incorporates these changes. The mapping table must therefore be updated in such a manner that the virtual address to the new data file points to this new copy of the data file, while the remaining virtual addresses point of the original data file location.

[0011] The problem of providing point in time copy semantics for the snapshot copy operation is solved by completing data file accesses that are already in progress before initiating the mapping table updates performed by the snapshot copy process and detecting accesses to the original data file or the copy file during the time that the snapshot copy process is being executed and the mapping table is being updated. The mapping table updates performed by the snapshot copy process are delayed until the completion of the data file accesses that preceded the command that initiated the snapshot copy process and any attempt to update the mapping table to reflect data written to the original data file or the copy data file that occurs after initiation of the snapshot copy process must wait until the first set of mapping table pointers have been updated. This ensures that the data file access operations behave as though data file access operations initiated before the snapshot copy operation were completed before the initiation of the snapshot copy operation and all data file access operations initiated after the initiation of the snapshot copy operation were initiated after the completion of the snapshot copy operation. The use of a fault tolerant cache allows this write delay to be hidden from the processor. Any request to read data from the copy data file received before the mapping table pointers have been updated is redirected to the original data file to ensure that the data file read operation behaves as though the snapshot copy process had been completed.

[0012] Thus, the present data file storage management system for snapshot copy operations both efficiently ensures the reliability of the mapping table data and also performs an incremental copy process which preserves the point in time copy semantics to ensure copy data file correspondence to the original data file.

#### Brief Description of the Drawing

##### [0013]

Figure 1 illustrates in block diagram form the architecture of a typical data storage subsystem in which the present data file storage management system for snapshot copy operations is implemented; Figure 2 illustrates in block diagram form the implementation of the two level mapping table as a Virtual Track Table and a Track Number Table; Figure 3 illustrates in block diagram form a typical implementation of a Logical Cylinder; Figure 4 illustrates in block diagram form a Logical Cylinder Directory entry which describes a Virtual Track Instance; Figure 5 illustrates in block diagram form a Logical Cylinder Directory entry which describes a Virtual Track Table Page Instance; Figures 6A - 6B illustrate in flow diagram form the operational steps taken by the controller to perform a snapshot copy process;

5 Figures 7A - 7B illustrate in flow diagram form the operational steps taken by the controller to carry out the free space collection process;

10 Figures 8-13 illustrate in block diagram form the state of the mapping table at various stages of a snapshot copy operation;

15 Figure 14 illustrates in flow diagram form the operational steps taken by the controller when screening data file read requests;

20 Figure 15 illustrates in flow diagram form the operational steps taken by the controller when screening virtual track write requests; and

25 Figures 16A - 16C illustrate in flow diagram form the operational steps taken by the present data file storage management system for snapshot copy operations to store a received data file and create copies thereof.

#### Detailed Description

20 [0014] The present data file storage management system for snapshot copy operations maintains a two level mapping table which enables the data files to be copied using the snapshot copy process but only requires the update of a single corresponding mapping table entry when the physical location of the data file instance is changed. The snapshot copy updates to the contents of the first level of the two level mapping table are stored on the backend data storage devices to provide a record of the snapshot copy operation which can be used to recover the correct contents of the mapping table. This record of the snapshot copy operation remains valid even though the physical location of a copied data file instance is subsequently changed. Furthermore, the physical storage space holding the updated portions of the first level of the two level mapping table can be managed using techniques like those used to manage the physical storage space holding data file instances. At least one of these two mapping tables are stored on the backend data storage devices to provide a backup of the mapping data that is stored in the virtual track directory. In addition, the synchronization of the snapshot copy operation with the reading and writing of data to the original and copy data files is maintained by detecting accesses to the original data file or the copy file during the time that the snapshot copy process is being executed and the mapping table is being updated. Mapping table updates resulting from the snapshot copy operation are delayed until all mapping table updates resulting from earlier data file write operations have been completed and any attempt to update the mapping table to reflect data written to the original data file or the copy data file that occurs after initiation of the snapshot copy operation must wait until the first set of mapping table pointers have been updated.

### Data Storage Subsystem Architecture

[0015] The present data file storage management system for snapshot copy operations 106 is implemented in a dynamically mapped virtual data storage subsystem 100 which is connected to at least one processor 101 and which functions to store data files for the processor 101 in the backend data storage devices 105 that are part of the data storage subsystem 100. The processor transmits a data file for storage in the data storage subsystem 100 over a selected one of the data channels 103 which serve to interconnect the processor 101 with the data storage subsystem 100. The processor 101 assigns a virtual track identification to the data file transmitted to the data storage subsystem 100 and the received data file is stored in an allocated physical storage location in the backend data storage devices 105 contained in the data storage subsystem 100. The assignment of a physical storage location on the backend data storage devices 105 is effected by a controller 104 contained within the data storage subsystem 100, which defines the correspondence between the processor 101 assigned virtual track identification and the logical address of the stored data file. This translation of the virtual track identification to the logical address corresponding to the physical storage location comprises the "dynamically mapped virtual" aspect of the data storage subsystem 100. A cache memory 102 is included in the data storage subsystem 100 to provide temporary storage for data files as well as data used by the controller 104. The present data file storage management system for snapshot copy operations 106 is implemented, in part, in the controller 104, with the exact implementation details of this system being a matter of engineering choice.

### Snapshot Copy Process

[0016] The snapshot copy process 107 is implemented by the use of a two level mapping architecture. The first level of mapping maps the processor 101 provided virtual address to an immutable name which identifies a unit of data, such as a virtual track. The second level of mapping maps the immutable name to a given logical address. The snapshot copy operation operates on the first level of the mapping table, thereby eliminating the need to associate with each virtual track address a mapping table entry which contains a logical address. The two mapping tables used to implement the two level mapping table, as shown in Figure 2, are the Virtual Track Table and the Track Number Table. The Virtual Track Table is indexed using the processor 101 provided virtual track address which then provides a unique immutable name, comprising a Track Number, that is used as the index for the Track Number Table. The Track Number is valid for the life of the virtual track instance, as determined by the reference count contained in the Track Number Table. The reference count represents the number of times a virtual track instance (physical

copy of the received data file) appears as a virtual track to the processor 101.

[0017] There are three sources for updates to the Virtual Track Table: snapshot copy, data file delete, and normal processor 101 write activity. When the processor 101 writes to an existing virtual track, the data file storage management system for snapshot copy operations 106 must check the reference count in the Track Number Table entry pointed to by the Track Number associated with that virtual track address. If the reference count is one, no other virtual tracks share the storage of the virtual track instance, therefore the new virtual track instance simply uses the same track number and the Track Number Table entry is updated to reflect the new logical address for this data file. The old logical address of the data file now contains invalid data (old virtual track instance) and the data storage subsystem 100 free space collection process eventually discards this old virtual track instance. If the processor 101 writes to a virtual track which has a reference count greater than one or writes to a track that has never been assigned a track number, a new track number must be assigned to the new virtual track instance. The data file storage management system for snapshot copy operations 106 decrements the reference count for the track number originally associated with the virtual track address (if any) to remove this virtual track's usage of the old virtual track instance. The data file storage management system for snapshot copy operations 106 then acquires an unused Track Number, which is assigned to the virtual track's address in the Virtual Track Table. There is one Track Number for each possible virtual track instance that is defined by the present virtual device configuration. If processor 101 instructs data storage subsystem 100 to delete a virtual track, data file management system for snapshot copy operations 106 decrements the reference count for the track number associated with the Virtual Track Address (if any) to remove this virtual track's usage of the virtual track instance. Data file management system for snapshot copy operations 106 then removes the association between the Virtual Track Address and the Track Number which previously identified the virtual track instance selected by the Virtual Track Address.

[0018] The benefits of a two level mapping table are:

- 50 1. Snapshot copy operations are performed without having to write new instances of the virtual tracks involved in the copy.
- 55 2. Error recovery capability of the mapping table. A copy is made of the mapping table Virtual Track Table and this data is written to the disks of the backend data storage devices 105 as with other data. The mapping table data is thereby recoverable the same as virtual track instances.
- 5 3. The number of copies of a virtual track instance is determined solely by the size of the reference count field.

### Virtual Track Table

**[0019]** Figure 2 illustrates in block diagram form the implementation of a two level mapping table in the present data file storage management system for snapshot copy operations 106. The first level of the mapping table, comprising the Virtual Track Table, provides the mapping from the individual virtual address provided by the processor 101 to an immutable name which is used to index the corresponding entry in the second level of the two level mapping table. The second level of the mapping table, comprising the Track Number Table, provides the mapping from the immutable name stored in the Virtual Track Table to the logical address which identifies the physical storage location in the backend data storage devices 105 that contains the received virtual track instance.

**[0020]** In the example provided herein, the processor 101 stores virtual tracks, which are associated with a virtual device, in the data storage subsystem 100. The Virtual Track Address, assigned by processor 101 to a virtual track, identifies a particular track by assigning the track a virtual cylinder number and a virtual track number. For each virtual device defined by the processor 101, the data storage subsystem 100 stores a list of addresses which point to Virtual Track Table Pages (VTT Pages), each page containing a predetermined number (for example: 8192) of byte segments of memory. The physical storage for these Virtual Track Table Pages may be within cache memory 102, within controller 104, within data storage devices 105, or a combination of these locations as a matter of engineering choice.

**[0021]** These Virtual Track Table Pages each contain an entry for each virtual track within a 128 cylinder boundary of the virtual device. The number of Virtual Track Table Pages per virtual device is dependent on the maximum number of virtual cylinders that are defined in the virtual device's configuration. Also contained within each Virtual Track Table Page is data which defines the Logical Address of a copy of the Virtual Track Table Page comprising a Virtual Track Table Page instance which has been written on backend data storage devices 105 during the snapshot copy operation. This Logical Address identifies the physical storage location in the backend data storage devices 105 that contains the most recently written instance of the present Virtual Track Table Page.

**[0022]** Each of the Virtual Track Table Pages comprise a plurality of table entries, which are indexed by cylinder number and virtual track number within the virtual device (for example: Cyl 0, Trk 0). The table entries comprise a series of flags and a Track Number comprising a data entry of predetermined size, which contains the data required to access a corresponding entry contained in the Track Number Table. In the example provided herein, the Track Number comprises a 28 bit entity which contains three segments: Track Number Partition, Track Number Segment Index, Track Number Suffix.

The Track Number Partition selects a list of Track Number Table Page addresses. The Track Number Segment Index is 10 bits in size and is the index into the list of Track Number Table Page addresses for this track partition number. The Track Number Suffix is 10 bits in size and comprises the index within the Track Number Table Page. The Track Number therefore comprises a unique identifier which points to a single entry in the Track Number Table, which contains data that is used to identify the physical storage location in the backend data storage devices 105 that contains the received virtual track instance.

### Track Number Table

**[0023]** Figure 2 also illustrates, in block diagram form, the implementation of a Track Number Table in the present data file storage management system for snapshot copy operations 106. Conceptually, the Track Number Table is a linear table that maps each Track Number representing a virtual track instance to a logical address which identifies the physical storage location holding the virtual track instance on the backend data storage devices 105. In practice, the Track Number Table is organized much like the Virtual Track Table, with portions of the table segmented into the Track Number Table Pages. The Track Number Table is initially indexed with the Track Number Partition which selects a list of Track Number Table Page addresses. The Track Number Segment Index then selects the appropriate Track Number Table Page address from the list of Track Number Table Page addresses. A Track Number Table Page address points to a Track Number Table Page which contains a predetermined number (for example: 8192) of byte segments of memory. These Track Number Table Pages each contain an entry for each virtual track within a 1024 Track Number boundary. As with the Virtual Track Table, the physical storage for these Virtual Track Table may be within cache memory 102, within controller 104, within data storage devices 105, or a combination of these locations as a matter of engineering choice.

**[0024]** A plurality of the Track Number Table Pages are grouped as segments of a cache buffer block, with a plurality of the cache buffer blocks being used to define the Track Number Table. In practice, the Track Number Partition identifies the one of the plurality of cache buffer blocks that contains the Track Number Table Page of interest. The Track Number Segment Index provides the next layer of granularity and identifies the particular Track Number Table Page within this cache buffer block that is of interest. Finally, the Track Number Suffix defines the particular Track Number Table Page entry that corresponds to the requested virtual track instance stored on the backend data storage devices 105.

**[0025]** The particular implementation of the two level mapping table disclosed herein represents one implementation and other variations of this scheme can be

used for the same purpose. What is of note herein is that the Track Number represents an immutable name for the virtual track instance, such that the Virtual Track Table can be manipulated by the present data file storage management system for snapshot copy operations 106 to perform the snapshot copy function without regard for the logical address of the virtual track instance. The logical address of the virtual track instance is managed such that any change in the physical storage location of the virtual track instance need be recorded only in a single location in the Track Number Table, regardless of the number of virtual track addresses associated with the virtual track instance in the Virtual Track Table.

#### Additional Data Structures

[0026] Data storage subsystem 100 formats the data it stores on backend data storage devices 105 in a data structure known as a Logical Cylinder. Figure 3 illustrates in block diagram form a typical implementation of a Logical Cylinder. The Logical Cylinder consists of a plurality of Virtual Track Instances and/or VTT Page Instances and a Logical Cylinder Directory (LCD). The Logical Cylinder Directory consists of a LCD entry for each Virtual Track Instance and/or VTT Page Instance in the Logical Cylinder, a count of the total number of LCD Entries in the LCD, a count of those LCD Entries which describe VTT Page Instances and a Logical Cylinder Sequence Number (LCSN). An optional pad area may separate the Virtual Track Instances and/or VTT Page Instances from the Logical Cylinder Directory so as to ensure that the final portion of the LCD is stored at the end of the regions of the backend data storage devices 105 which together are used to hold the Logical Cylinder. The LCD summarizes the contents of the Logical Cylinder. The Logical Cylinder Sequence Number can be used to determine the order in which Logical Cylinder were written, enabling the recovery of the mapping table by repeating the sequence of mapping table updates which took place when each logical cylinder was written.

[0027] Figure 4 depicts in block diagram form an Logical Cylinder Directory entry which describes a Virtual Track Instance. Each Logical Cylinder Directory entry contains an Logical Cylinder Directory Entry Type, which in this case identifies the entry as being one which describes a Virtual track Instance. The Virtual Track Address field within this type of Logical Cylinder Directory entry contains the Virtual Track Address which processor 101 initially assigned to the virtual track when it was most recently written or modified. If a virtual track and then the physical position of the virtual track instance is changed by the free space collection process, a NULL value is written into the Virtual Track Address field. This NULL value indicates that even though the Virtual Track Instance still represents the current data file associated with the Virtual Track Address to which the data file was most recently written.

[0028] The Track Number field contains the immutable name by which this Virtual track Instance is known. The Track Number contained in the Track Number field of this type of Logical Cylinder Directory entry is used to 5 select the entry in the Track Number Table which contains the logical address of the Virtual Track Instance described by the Logical Cylinder Directory entry.

[0029] The Logical Address and Virtual Track Instance Length fields identify where in the Logical Cylinder the Virtual Track Instance starts and ends.

[0030] Figure 5 depicts in block diagram form a Logical Cylinder Directory entry which describes a Virtual Track Table Page Instance. The Logical Cylinder Directory Entry Type identifies the Logical Cylinder Directory entry as being one which describes a VTT Page Instance. The VTT Page Identifier field identifies which

15 VTT Page is contained in the Logical Cylinder. The Original LCSN field contains the LCSN of the Logical Cylinder in which this instance of the VAT Page was first written. This will differ from the LCSN of the Logical Cylinder that currently contains the VTT Page Instance when the physical location of the VAT Page Instance has been 20 changed by the free space collection process. Preserving the LCSN of the Logical Cylinder in which the VTT Page Instance was originally written allows the recovery of the mapping table updates performed by a snapshot copy operation when the mapping table is being recovered by repeating the sequence of mapping table updates which took place when each logical cylinder was

25 written. In the case of the mapping table updates captured in the VTT Page Instance, the time at which the mapping table recovery process reads the VTT Page Instance into the mapping table is determined by the LCSN of the logical cylinder into which the VTT Page Instance was originally written, not the LCSN of the Logical Cylinder in which the VTT Page Instance is currently stored.

[0031] Figure 6 illustrates in flow diagram form the operational steps taken by controller 104 to carry out snapshot copy process 107 in response to processor 101 instructing data storage subsystem 100 to copy one or 30 more source data files for which processor 101 has assigned one or more Virtual Track Addresses (these data files constituting the source area) to an equal number of target data files which are again identified by processor 101 using one or more Virtual Track Addresses (the target area). Host processor 101 has the flexibility to specify a single source data file multiple times in its definition of the source area. In this fashion, processor 101 may

35 45 employ a single snapshot copy command issued to data storage subsystem 100 to direct the snapshot copy process 107 depicted in Figure 6 to make a plurality of copies of a source data file.

[0032] At step 601, controller 104 receives the identity 50 of the copy source data files from processor 101. In preferred implementation, processor 104 specifies the data files to be copied by transmitting over data channel 103 the Virtual Track Addresses of the data files (also known

as virtual tracks or simply tracks) to be copied. At step 602, controller 104 determines whether modified forms of any of the source tracks are stored in cache memory 102. If such modified source tracks are found in cache memory 102, processing proceeds with step 603 at which point controller 104 writes the modified source tracks to backend data storage devices 105. If processor 100 had issued a command to write data to a source track before the initiation of the snapshot copy command and this write command has not yet completed, the processor 101 initiated write to the source track is allowed to complete before the modified form of the source track is written to backend data storage devices 105. At step 604 controller 104 updates the Track Number Table (TNT) by writing into the TNT entries which describe the virtual track instances written at step 603 the logical addresses of the locations within backend data storage devices 105 at which the modified track instances were stored. At the completion of step 604 or if no modified source tracks were found in cache memory 102 at step 602, processing proceeds with step 605.

[0033] At step 605, controller 104 receives the identity of the copy target data files from processor 101. As in the case of receiving the identity of the copy source data files, processor 104 transmits over data channel 103 the Virtual Track Addresses of the copy target data files. The copy target data files are the ones to which the copy source data files will appear to have been copied as soon as the specification of the copy target data files has been received by controller 104.

[0034] At step 606, controller 104 determines whether any of the target tracks are stored in cache memory 102. If target tracks are found in cache memory 102, processing proceeds with step 607 at which point controller 104 removes the target tracks from cache memory 102. If processor 100 had issued a command to read data from a target track before the initiation of the snapshot copy command and this read command has not yet completed, the target track read command is allowed to complete before the target track being read is removed from cache memory 102. The target tracks removed from cache memory 102 represent the data files which processor 101 had written into the copy target area prior to the initiation of the snapshot copy operation. Removing these target tracks from cache memory 102 ensures that if processor 101 reads a target data file from data storage subsystem 100 after it has transmitted the definition of the target area to storage subsystem 100, processor 101 will not receive the data files which were stored in the target data area prior to the snapshot copy operation. At the completion of step 607 or if no target tracks were found in cache memory 102 at step 606, processing proceeds with step 608.

[0035] At step 608, controller 104 chooses a Virtual Track Table Page which contains one or more entries that are selected by target Virtual Track Addresses. Step 608 is the first step in a program loop consisting of steps

608 through 618. Each iteration of this program loop carries out the steps required to update a Virtual Track Table Page and store the Virtual Track Table Page on backend data storage devices 105.

- 5 [0036] At step 609, controller 104 chooses a Virtual Track Table entry within the currently selected Virtual Track Table Page which maps a target Virtual Track Address to a Track Number. Step 609 is the first step in a program loop consisting of steps 609 through 615. Each iteration of this program loop carries out the steps required to update a Virtual Track Table entry so that the Virtual Track Table entry maps a target Virtual Track Address to the same Track Number as the Track Number which represents the source data file which is being copied to the target Virtual Track Address.
- 10 [0037] At step 610, controller 104 determines whether the Track Number contained in the selected Virtual Track Table entry contains a NULL value. If no data has ever been written to the data file described by the selected Virtual Track Table entry or processor 101 has instructed data storage subsystem 100 to delete the data file described by the selected Virtual Track Table entry, the Track Number field in the Virtual Track Table entry will contain a NULL value. If the Track Number stored
- 15 in the selected Virtual Track Table entry is found not to be a NULL value, processing proceeds with step 611. At step 611, controller 104 decrements the reference count in the TNT entry selected by the Track Number stored in the selected Virtual Track Table entry. This indicates that there is now one less Virtual Track Address by which the data file described by the TNT entry can be accessed. If the resulting reference count is zero, it indicates that the data file described by the TNT entry can no longer be accessed by any Virtual Track Address.
- 20 In this case, the virtual track instance described by the TNT entry is no longer needed and the Track Number which selects the TNT entry is free to be used to describe some new data file to which processor 101 will assign some other Virtual Track Address. At the completion of step 611 or if the Track Number in the currently selected Virtual Track Table entry was found to be NULL at step 610, processing proceeds with step 612.
- 25 [0038] At step 612, controller 104 copies the Track Number from the Virtual Track Table entry selected by the source Virtual Track Address to the Virtual Track Table entry selected by the current target Virtual Track Address. This will cause processor 101 reads from either the source Virtual Track Address or the Target Virtual Address to read the virtual track instance described by the Track Number which was just copied.
- 30 [0039] At step 613, controller 104 determines whether the source Track Number which was just copied is a NULL value. If the source Track Number is found not to be a Null value, processing proceeds with step 614. At
- 35 step 614, controller 104 increments the reference count in the TNT entry selected by the source Track Number. This indicates that there is now one more Virtual Track Address by which the data file described by the TNT en-

try can be accessed. At the completion of Step 614 or if the source Track Number was found to be NULL at step 613, processing proceeds with step 615.

[0040] At step 615, controller 104 determines whether there are additional target tracks within the currently selected Virtual Track Table Page which have not yet been processed. If there is an as yet unprocessed target track which selects a Virtual Track Table entry within the current Virtual Track Table Page, processing proceeds with step 609. Otherwise, processing proceeds with step 616.

[0041] At step 616, controller 104 writes the contents of the current Virtual Track Table Page to the backend data storage devices 105. This creates a new Virtual Track Table Page Instance within a Logical Cylinder. When writing a Virtual Track Table Page instance to a Logical Cylinder, controller 104 also writes a Virtual Track Table page instance Logical Cylinder Directory Entry to the Logical Cylinder Directory within the Logical Cylinder. This Logical Cylinder Directory entry describes the Virtual Track Table Page instance stored within the Logical Cylinder and contains the same Logical Cylinder Sequence Number which is contained in the Logical cylinder Directory. The LCSN field of the Virtual Track Table Page Instance Logical Cylinder Directory Entry identifies the Logical cylinder in which the Virtual Track Table Page instance was first written so that even if the Virtual Track Table Page instance is subsequently moved to a different logical cylinder, the mapping table can be recovered by repeating the sequence of mapping table updates which took place when each logical cylinder was written.

[0042] At step 617, controller 104 writes into the Logical Address field of the Virtual Track Table Page within the mapping table which was written at step 616 the logical address at which the Virtual Track Table Page instance was written at step 616.

[0043] At step 618, controller 104 determines whether there are additional Virtual Track Table Pages which contain entries that are selected by target tracks that have not yet been processed. If there is an additional Virtual Track Table page containing a Virtual Track Table entry which describes an as yet unprocessed target track, processing proceeds with step 608. Otherwise, processing proceeds with step 619.

[0044] At step 619, controller 104 transmits status information over data channel 103 indicative of the completion of the snapshot copy operation.

[0045] Figure 7 illustrates in flow diagram form the operational steps taken by controller 104 to carry out the free space collection process which is a part of data file management system for snapshot copy operations 106. The free space collection process moves virtual track instances and Virtual Track Table Page instances from one logical cylinder to another in order to create completely empty logical cylinders into which new virtual track instances and Virtual Track Table Page instances can be written. When the free space collection process

encounters a virtual track instance or a Virtual Track Table Page instance, it determines whether controller 104 has written an updated virtual track instance or an updated Virtual Track Table Page instance to some other physical location on backend data storage devices 105. When a more recent instance exists on backend data storage devices 105, the free space collection process does not copy the now obsolete instance to a different logical cylinder. In this way, the free space collection process discards virtual track instances which are now obsolete because processor 101 has updated the data file contained within the virtual track instance and controller 104 has written this updated data file to a different location in backend data storage devices 105. Similarly, the free space collection process discards Virtual Track Table Page instances which are now obsolete because a later snapshot copy operation has caused a more recent instance of the Virtual Track Table Page to be written to a different location in backend data storage devices 105. Thus, it is the free space collection process which limits the amount of memory contained in backend data storage devices 105 which is consumed holding the record of snapshot copy operations. This record of snapshot copy operations may be used by controller 104 to allow it to recover the mapping table. The free space collection process discards obsolete Virtual Track Table page instances, resulting in there being no need to store more Virtual Track Table page instances on backend data storage devices 105 than the number of Virtual Track Table Pages within the mapping table. Without some process for discarding the record of an old snapshot copy operation, an unbounded amount of memory within backend data storage devices 105 would be consumed as processor 101 issues an unlimited number of snapshot copy commands.

[0046] At step 701, controller 104 selects a logical cylinder which will have its contents moved to a different logical cylinder so that the selected logical cylinder will be made empty and available for holding new virtual track instances and Virtual Track Table Page instances.

[0047] At step 702, controller 104 determines whether there is another Logical Cylinder Directory entry in the Logical Cylinder Directory for the selected logical cylinder which has not yet been processed. Step 702 is the first step in a program loop consisting of steps 702 through 724. Each iteration of this program loop carries out the steps required to determine whether the virtual track instance or the Virtual Track Table Page instance described by the currently selected Logical Cylinder Directory entry should be moved to another logical cylinder. If the instance should be moved to another logical cylinder, the steps within this program loop write the instance to the other logical cylinder, creating a Logical Cylinder Directory entry which will be written to the other logical cylinder. If, at step 702, controller 104 determines that there are no more Logical Cylinder Directory entries in this logical cylinder which have yet to be processed, processing proceeds with step 703. Step 703

marks the currently selected logical cylinder as being empty so that the memory area within data storage devices 105 which holds the logical cylinder is available for being used to store new virtual track instances and new Virtual Track Table Page instances. When step 703 completes, processing proceeds with step 701.

[0048] If controller 104 determines at step 702 that there is an additional Logical Cylinder Directory entry within this logical cylinder which is yet to be processed, processing proceeds with step 704. At step 704, controller 104 selects the next Logical Cylinder Directory entry to be processed. Then at step 705, controller 104 examines the Logical Cylinder Directory Entry Type field of the Logical Cylinder Directory entry to determine whether the Logical Cylinder Directory entry describes a virtual track instance or a Virtual Track Table Page instance. If the Logical Cylinder Directory entry describes a virtual track instance, processing proceeds with step 706.

[0049] At step 706, controller 104 determines whether the reference count field of the TNT entry selected by the Track Number in the Logical Cylinder Directory entry is non-zero. If the reference count field contains the value zero, the virtual track instance described by this Logical Cylinder Directory entry is no longer needed because processor 101 has instructed data storage subsystem 100 to delete the data file contained in the virtual track instance selected by the Track Number in the current Logical Cylinder Directory entry. In this case, there is no need to preserve the virtual track instance described by the current Logical Cylinder Directory entry by writing the virtual track instance to another logical cylinder so processing proceeds with step 702. If controller 104 determines at step 706 that the reference count field of the TNT entry selected by the Track Number in the Logical Cylinder Directory entry is non-zero, processing proceeds with step 707.

[0050] At step 707, controller 104 determines whether the logical address contained in the TNT entry selected by the Track Number in the current Logical Cylinder Directory entry is equal to the logical address of virtual track instance described by the current Logical Cylinder Directory entry. If the two logical addresses are not equal, the virtual track instance described by the current Logical Cylinder Directory entry is obsolete because a more recent instance of this virtual track is stored at a different logical address than the virtual track instance described by this Logical Cylinder Directory entry. In this case, there is no need to preserve the virtual track instance described by the current Logical Cylinder Directory entry by writing the virtual track instance to another logical cylinder so processing proceeds with step 702. If controller 104 determines at step 707 that the two logical addresses are equal, the virtual track instance described by the current Logical Cylinder Directory entry is the most recent instance of this virtual track that is stored within backend data storage devices 105 and processing proceeds with step 708.

[0051] At step 708, controller 104 determines whether the Virtual Track Address field in the current Logical Cylinder Directory entry contains a NULL value. If the Virtual Track Address field does not contain a NULL value,

- 5 processing proceeds with step 709 where controller 104 determines whether or not the Track Number in the Virtual Track Table entry selected by the Virtual Track Address field of the current Logical Cylinder Directory entry is equal to the Track Number contained in the current Logical Cylinder Directory entry. If the two track numbers are not equal, processor 101 has updated the data file stored within the virtual track instance but because the data file had been copied to another Virtual Track Address, a new Track Number was assigned to represent the modified data file. In this case, the virtual track instance described by the current Logical Cylinder Directory entry can no longer be accessed using the Virtual track Address at which it was originally written by processor 101 and processing proceeds with step 710.
- 10 Also, if controller 104 determines at step 708 that the Virtual Track Address in the current Logical Cylinder Directory entry already contains a NULL value, processing proceeds with step 710.

[0052] At step 710, controller 104 writes a NULL value into the Virtual Track Address field of the new Logical Cylinder Directory entry which will describe the current virtual track instance when it is written to another logical cylinder.

- 15 [0053] If controller 104 determines at step 709 that the Virtual Track Address in the current Logical Cylinder Directory entry is mapped to the same Track Number as is stored in the Track Number field of the current Logical Cylinder Directory entry, processing proceeds with step 711. At step 711, controller 104 copies the Virtual Track Address stored in the current Logical Cylinder Directory entry into the new Logical Cylinder Directory entry.
- 20

[0054] After the completion of either step 710 or step 711, processing proceeds with step 712. At step 712, controller 104 copies the Track Number from the current Logical Cylinder Directory entry to the new Logical Cylinder Directory entry. Then, at step 713, controller 104 copies the virtual track instance length from the current Logical Cylinder Directory to the new Logical Cylinder Directory.

- 25 [0055] At step 714, controller 104 writes to another logical cylinder within backend data storage devices 105 the virtual track instance described by the current Logical Cylinder Directory entry. At step 715, controller 104 writes into the Logical Address field of the new Logical Cylinder Directory entry the logical address at which the virtual track instance was written at step 714. This completes the process of moving the virtual track instance described by the current Logical Cylinder Directory entry to another logical cylinder. After the completion of step 717, processing proceeds with step 702. At step 714, controller 104 writes to another logical cylinder within backend data storage devices 105 the virtual track instance described by the current Logical Cylinder Direc-
- 30
- 35
- 40
- 45
- 50
- 55

tory entry. At step 715, controller 104 writes into the Logical Address field of the new Logical Cylinder Directory entry the logical address at which the virtual track instance was written at step 714. At step 716, controller 104 writes the new Logical Cylinder Directory entry into the Logical Cylinder Directory contained in the logical cylinder to which the virtual track instance was written at step 714.

[0056] At step 717, controller 104 writes into the TNT entry selected by the Track Number contained in the current Logical Cylinder Directory entry the logical address at which the virtual track instance was written at step 714. This completes the process of moving the virtual track instance described by the current Logical Cylinder Directory entry to another logical cylinder. After the completion of step 717, processing proceeds with step 702.

[0057] If controller 104 determines at step 705 that the current Logical Cylinder Directory entry describes a Virtual Track Table Page instance processing proceeds with step 718.

[0058] At step 718, controller 104 uses the Virtual Track Table Page Identifier in the current Logical Cylinder Directory entry to select a Virtual Track Table Page within the mapping table. Controller 104 then reads the Logical Address field from this Virtual Track Table page within the mapping table and compares this logical address to the logical address of the Virtual Track Table Page instance described by the current Logical Cylinder Directory entry. If the two logical addresses are not equal, the Virtual Track Table Page instance described by the current Logical Cylinder Directory entry is obsolete because a later snapshot copy operation caused a more recent instance of this Virtual Track Table Page to be stored at a different logical address than the Virtual Track Table Page instance described by this Logical Cylinder Directory entry. In this case, there is no need to preserve the Virtual Track Table Page instance described by the current Logical Cylinder Directory entry by writing the Virtual Track Table Page instance to another logical cylinder so processing proceeds with step 702. If controller 104 determines at step 718 that the two logical addresses are equal, the Virtual Track Table Page instance described by the current Logical Cylinder Directory entry is the most recent instance of this Virtual Track Table Page that is stored within backend data storage devices 105 and processing proceeds with step 719.

[0059] At step 719, controller 104 copies the Virtual Track Table Page Identifier stored in the current Logical Cylinder Directory entry into the new Logical Cylinder Directory entry which will describe the current Virtual Track Table Page instance when it is written to another logical cylinder.

[0060] At step 720, controller 104 copies the contents of the Original LCSN field from the current Logical Cylinder Directory entry to the new Logical Cylinder Directory entry.

[0061] At step 721, controller 104 writes to another

logical cylinder within backend data storage devices 105 the Virtual Track Table Page instance described by the current Logical Cylinder Directory entry. At step 722, controller 104 writes into the Logical Address field of the new Logical Cylinder Directory entry the logical address at which the Virtual Track Table Page instance was written at step 721. At step 723, controller 104 writes the new Logical Cylinder Directory entry into the Logical Cylinder Directory contained in the logical cylinder to

5 which the Virtual Track Table Page instance was written at step 721.

[0062] At step 724, controller 104 writes into the Logical Address field of the Virtual Track Table Page within the mapping table selected by the Virtual Track Table

10 Page Identifier field of the current Logical Cylinder Directory entry the logical address at which the Virtual Track Table Page instance was written at step 721. This completes the process of moving the Virtual Track Table Page instance described by the current Logical Cylinder

15 Directory entry to another logical cylinder. After the completion of step 724, processing proceeds with step 702.

[0063] The free space collection process is an ongoing process within data file management system for snapshot copy operations 106. The free space collection

20 process will pause at step 701 if there are no logical cylinders which contain obsolete virtual track instances or obsolete Virtual Track Table Page instances or if the free space collection process determines that data storage subsystem 100 would not benefit from the immediate collection of the free space from an additional logical cylinder. The details of this aspect of the behavior of the free space collection process are not relevant to the understanding of the present invention.

25 **Virtual Track Management Example**

[0064] Figures 8 -13 illustrate, in block diagram form, the state of the mapping table and particular logical cylinders at various stages of a snapshot copy operation,

40 which is also illustrated in flow diagram form in Figures 16A - 16C. This represents a chronological view of a set of virtual tracks and their associated data structures from their conception through a number of typical snapshot copy and data storage subsystem 100 operations.

45 For the purposes of simplicity of description, the representation for the various entries in the two level mapping table and the logical cylinders are simplified to thereby illustrate the concepts of the present system and avoid the complexity of the implementation details.

[0065] The virtual track is created within the processor 101 as part of the processor 101's data management function. The processor 101 is connected to and views the dynamically mapped virtual data storage subsystem 100 as a particular size and architecture data storage

50 device. Therefore, when the processor 101 wishes to store a virtual track instance on the "data storage device" embodied by the data storage subsystem 100, the processor 101 formats the virtual track instance and as-

signs it a particular virtual track address. The processor 101 then transmits the virtual track instance and its associated virtual track address to the data storage subsystem 100 at step 1601 over a selected one of the data channels 103. In the present example, it is assumed that the processor 101 creates two virtual track instances for storage on the data storage subsystem 100. The data storage subsystem 100, upon receipt of these two virtual track instances at step 1602, stores this data in cache memory 102 and the controller 104 allocates two logical addresses which identify two physical data storage locations on the backend data storage devices 105 for the storage of the two received virtual track instances at step 1603. The controller 104 also generates two immutable names for these two virtual track instances at step 1604, with the immutable names comprising Track Numbers as described above. The controller 104 then writes the two virtual track instances from the cache memory 102 to the allocated logical addresses where the data is stored at step 1605. The controller 104 at step 1605 then updates the two level mapping table to reflect the presence of these two virtual track instances on the backend data storage devices 105. The mapping table entries comprise two entries in the Virtual Track Table VTT for Device N, Cylinder X, with the first entry VT#1 having an entry of TN=1 and the second entry VT#2 having an entry TN=2. These Track Numbers (TN=1, TN=2) are shortened versions of the actual numbers but serve to illustrate the operation of the mapping table. The Track Numbers point to corresponding entries in the Track Number Table TNT, wherein data indicative of the logical address and reference count for the two virtual track instances are stored. The logical address data for the first and second virtual track instances are LA=0.1.0 and LA=0.1.1, respectively, indicative of the physical storage locations of logical device 0, logical cylinder 1, sectors 0, 1, respectively. Since these two virtual track instances are newly created and no other virtual track addresses point to them, the reference counts RC are set to 1 to indicate that a single virtual track address points to the respective virtual track instances. Thus, as shown in Figure 8, at the end of time period 1, the processor 101 has created two virtual tracks, virtual track 1 and virtual track 2, both of which are stored in the allocated logical addresses comprising logical cylinder 1 of logical device 0, sectors 0 and 1. The Logical Cylinder Directory LCD for logical cylinder 1 contains data entries that identify the existence of virtual tracks 1, 2. The logical address entry in the Virtual Track Table Page contains a NULL value, since no snapshot copies have been made involving virtual tracks in the page.

[0066] At step 1607, the processor 101 requests the creation of a snapshot copy of virtual tracks 1, 2 and has assigned Virtual Track Addresses of N.X.3 and N.X.4 respectively to these two copies of the original virtual track instances. The Virtual Track addresses N.X.3 and N.X.4 are indicative of virtual device N, cylinder X, tracks

3, 4, respectively. In response to receipt of commands from the processor 101 received over a selected data channel 103, of step 1608 the data storage subsystem 100 activates the data file storage management system 5 for snapshot copy operations 106 to create a single copy of both virtual tracks. This is accomplished at step 1609 by simply creating two duplicative pointers to point to the same virtual track instances. The Virtual Track Table entries VT#1, VT#2 are copies to Virtual Track Table entries VT#3, VT#4, on the same virtual cylinder, respectively. At the end of time period 2, the reference counts RC in the Track Number Table have been incremented to 2, to indicate that there are two Virtual Track Table entries VT#1, VT#3, and VT#2, VT#4 pointing to the virtual track instances V1, V2 respectively, stored in the physical memory. These two virtual track instances are identified as V1, V2 only for the purposes of this illustration to allow later instances of the same virtual tracks to be identified uniquely. In its operation, data file storage 10 management system for snapshot copy operations 106 does not utilize or store names such as V1, V2 to uniquely identify particular virtual track instances. Instead, the particular virtual track instance which represents the data most recently written to a Virtual Track Address is 15 identified by the Track Number which points to the virtual track instance. The data for mapping a Virtual Track Address to a Track Number is stored in the Virtual Track Table entries VT#3, VT#4, with the entries stored therein comprising the Track Numbers TN=1, TN=2 as with the 20 entries for the original Virtual Track Addresses, since the virtual tracks are unmodified. A Virtual Track Table Page Logical Cylinder Directory entry is made in the Logical Cylinder Directory. This entry in the Logical Cylinder Directory contains the identify of the Virtual Track Table Page and the Logical Cylinder Sequence Number assigned to the current instance of logical cylinder A. Also, the Virtual Track Table Page that contains the virtual tracks that were affected by the snapshot copy operation is written to the logical cylinder in the same manner 25 as virtual track instances. The backup of the Virtual Track Table Page contained within logical cylinder A allows a mapping table recovery process to reconstruct the mapping tables without requiring that either the copied virtual track instances be rewritten or the complete history of all copy operations be maintained on backend data storage devices 105. The backup of an entire page is a fundamental concept in the two level mapping table design in that it simplifies the free space collection process by maintaining the snapshot copy history in Virtual 30 Track Table Page size units rather than virtual track size units. Therefore, only the most recent version of that page needs to be maintained on the backend data storage devices 105 even though multiple copies within that page may have occurred. 35

[0067] At step 1610, the processor 101 updates or modified virtual track 1 in cylinder X of device N and the modified version of virtual track 1 must now be embodied in a physical virtual track instance which differs from 40

the original virtual track instance of virtual track 1 to reflect these changes. The data storage subsystem 100 creates a new virtual track instance at step 1611 by applying the modifications received from processor 101 to a physical copy of the original virtual track instance V1, resulting in a new virtual track instance which, for purposes of illustrating this example, is identified in Figure 10 as virtual track instance V4. The data storage subsystem 100 writes the modified version of virtual track instance V1, as virtual track instance V4, onto the backend data storage device location logical device 0, cylinder 2, sector 2 (LA=0.2.2) and must update the mapping table to reflect the presence of a new virtual track instance at step 1612. In particular, the data file storage management system for snapshot copy operations 105 generates an immutable name for the new virtual track instance with the immutable name comprising a Track Number as described above. The data file storage management system for snapshot copy operations 106 then updates the two level mapping table to reflect the presence of this new virtual track instance on the backend data storage devices 105. The mapping table entries comprise an entry in the Virtual Track Table for virtual device N, cylinder X, with the first entry VT#1 being converted to an entry of TN=3 to reflect the fact that the current virtual track instance selected by this Virtual Track Address is a modified version of the original virtual track instance but the original virtual track instance must be preserved because it can be accessed by a different Virtual Track Address. Virtual track instance V1 no longer represents the contents of virtual track 1 with the original data at sector 0 now being pointed to only by the Track Number Table entry that is pointed to by the Virtual Track Table entry for virtual track 3. At the end of time period 3, the reference count in the Track Number Table entry has been decremented and a new entry (3) has been added to the Track Number Table. The Logical Cylinder Directory for logical cylinder 2 contains a normal virtual track entry for the updated virtual track.

[0068] At step 1613, the processor 101 requests the creation of a snapshot copy, with virtual tracks 1 and 2 being copied to virtual tracks 5 and 6 respectively, of the same cylinder. The controller 104 responds to receipt of this snapshot copy request by creating two new entries in the mapping table to reflect the snapshot copy of two virtual tracks. In particular, the data file storage management system for snapshot copy operations 106 duplicates the two immutable names that were used for the two original virtual track instances at step 1614, with the immutable names comprising the two originally created Track Numbers TN=2 and TN=3 as described above. The data file management system for snapshot copy 106 then updates the two level mapping table to reflect the association of these new virtual track addresses with the already existing virtual track instances at step 1616. The mapping table entries comprise two entries in the Virtual Track Table for device N, cylinder X, with the first entry VT#5 being populated with an entry of TN=3 to

reflect the fact that the data stored at this virtual track address is the data which resulted from the modifications written to virtual track 1 at step 1610 above and the second entry, VT#6 being populated with an entry of TN=2 to reflect the fact that this virtual track address selects the virtual track instance which was originally written to virtual track 2. The reference counts for the Track Number Table entries 2 and 3 have been incremented to RC=3 and RC=2 to reflect the current number of Virtual Track Table entries which now point to these Track Number Table entries and thence to the virtual track instances V4 and V2 which reside on the backend data storage devices 105. A Virtual Track Table Logical Cylinder Directory entry is made in the Logical Cylinder Directory which contains the identify of the Virtual Track Table Page and the Logical Cylinder Sequence Number assigned to the current instance of logical cylinder C. The Virtual Track Table Page that contains the virtual tracks which were affected by the snapshot copy is written to logical cylinder C. Figure 11 illustrates the state of the mapping table at the end of time period 4 at which point the data originally written to virtual track 1 at step 1601 now exists as a single instance of data addressed only by virtual track 3 as a consequence of the first snapshot copy that was performed in this sequence and the subsequent writing of modified data to virtual track 1. [0069] At step 1622, the controller initiates the free space collection process which collects logical cylinders 1 and A, creating logical cylinder D, which contains all of the collected area. Logical cylinder 1 contains Logical Cylinder directory entries that define the creation of virtual track instances V1 and V2 which processor 101 originally wrote to Virtual Track Addresses N.X.1 and N.X.2. Logical cylinder A contains a Logical Cylinder Directory entry which describes the Virtual Track Table Page that defines the first snapshot copy operation which was performed at step 1609. For virtual track entries, the free space collection process uses the track number that is contained within the Logical Cylinder Directory entry to index the Track Number Table to determine whether the current virtual track instance is the most recent. In this case, at the start of time period 5, the Track Number Table entries for track numbers 1 and 2 each contain a non-zero reference count and the logical addresses within each of these two entries in the Track Number Table are equal to the logical addresses of the virtual track instances V1 and V2 within logical cylinder 1. Therefore, the two virtual track instances V1 and V2 can each be accessed by one or more Virtual Track Addresses and these two virtual track instances are still the most recent instances associated with track numbers 1 and 2 respectively. The free space collection process copies the virtual track instances V1 and V2 from logical cylinder 1 to logical cylinder D so that logical cylinder 1 can be used to hold new virtual track instances and Virtual Track Table Page instances. Before writing the Logical Cylinder Directory entries describing virtual track instances V1 and V2 to the Logical Cylinder

Directory within logical cylinder D, the free space collection process determines whether these virtual track instances are still selected when processor 101 accesses the Virtual Track Address at which the virtual track instances were originally written. In the case of virtual track instance V1, the free space collection process uses the Virtual Track Address X.N.1 from the Logical Cylinder Directory entry for virtual track instance V1 to index the Virtual Track Table. This Virtual Track Address selects Virtual Track Table entry VT#1 which, at step 1622 contains TN=3. The track number in Virtual Track Table entry VT#1 is no longer equal to the track number in the Virtual Track Instance Logical Cylinder Directory entry for virtual track instance V1. Therefore, the free space collection process writes a NULL value into the Virtual Track Address field of the new Logical Cylinder Directory entry for virtual track instance V1 that it writes to logical cylinder D. However, in the case of virtual track instance V2, the Virtual Track Table entry VT#2 selected by Virtual Track Address X.N.2 still contains TN=2. Therefore, the free space collection process copies the virtual track address X.N.2 from the Logical Cylinder Directory entry for virtual track instance VT2 in logical cylinder 1 into the new Logical Cylinder Directory entry for virtual track instance VT2 in logical cylinder D. The free space collection process then writes into the Track Number Table entries for Track Numbers 1, 2 the logical addresses identifying the physical locations within logical cylinder D of virtual track instances V1, V2, respectively.

[0070] For Virtual Track Table Page Logical Cylinder Directory entries, the free space collection process uses the Virtual Track Table Page identifier contained in the entry to index into the Virtual Track Table to find the logical address of the most recent instance of the Virtual Track Table Page which was written to backend data storage devices 105. The logical address read from the selected Virtual Track Table Page is compared to the logical address of the Virtual Track Table Page instance described by the Virtual Track Table Page Logical Cylinder Directory entry. If the two logical addresses are not equal, then the Virtual Track Table Page described by the current Logical Cylinder Directory entry is not the most recent instance stored on backend data storage devices 105 and the Virtual Track Table Page instance described by the current Logical Cylinder Directory entry can be discarded. The logical address contained in the Virtual Track Table Page identifies a more recent instance of this Virtual Track Table Page which was written to backend data storage devices 105 as a consequence of a later snapshot copy operation. Thus, the free space collection process does not copy the Virtual Track Table Page instance V4 from logical cylinder C to logical cylinder D. Figure 12 shows the contents of the mapping table and the contents of logical cylinder D at the end of time period 5.

[0071] At step 1623, the controller 104 initiates the free space collection process which collects logical cylinder

5 cylinder C, creating logical cylinder E, which contains all of the collected data. Logical Cylinder C contains a Logical Cylinder Directory entry which describes the Virtual Track Table Page that defines the second snapshot copy operation which was performed by steps 1614 and 1615. The free space collection process uses the Virtual Track Table Page identifier contained in the Virtual Track Table Page Logical Cylinder Directory entry to index into the Virtual Track Table to find the logical address of the 10 most recent instance of the Virtual Track Table Page which was written to backend data storage device 105. In this case, the logical address stored in the Virtual Track Table Page is equal to the logical address of the Virtual Track Table Page instance described by the Logical Cylinder Directory in logical cylinder C. This means that the Virtual Track Table Page instance is the most recent instance of this Virtual Track Table Page stored on backend data storage devices 105 so the free space collection process copies the Virtual Track Table Page 15 instance from logical cylinder C to logical cylinder E. In the new Virtual Track Table Page Logical Cylinder Directory Entry written to logical cylinder E by the free space collection process, the Original Logical Cylinder Sequence Number field contains a copy of the Original 20 Logical Cylinder Sequence Number field from the Logical Cylinder Directory entry from logical cylinder C. Should it become necessary to recover the mapping table using the Logical Cylinder Directory entries, the Virtual Track Table Page Logical Cylinder Directory entry 25 will indicate that the appropriate time to apply the Virtual Track Table Page instance which the free space collection process copied to logical cylinder E is after applying the mapping table updates resulting from the writing of logical cylinders with a logical Cylinder Sequence Number less than 83 and before applying updates resulting from logical cylinders with a Logical Cylinder Sequence Number greater than 83. Now that the Virtual 30 Track Table Page instance has been copied from logical cylinder C to logical cylinder E, the free space collection process writes the logical address of the newly written Virtual Track Table Page instance into the Virtual Track Table Page.

#### Maintenance of Point in Time Copy Semantics

45 [0072] For a snapshot copy operation to provide point in time copy semantics, the data read and write operations initiated by processors 101-1 through 101-K must behave as though the data files being copied using the 50 snapshot copy method were copied at the instant the snapshot copy command is received by data storage subsystem 100. To provide this behavior, controller 104 within data storage subsystem 100 carries out steps to ensure that data access operations performed by any 55 processor 101 are completed before the mapping table updates resulting from the snapshot copy operation are performed. Steps 601 through 607 in Figure 6 ensure that all data written to the copy source area before the

initiation of the copy operation will appear in the copy target area, all data written to the copy target area before the initiation of the copy will be overwritten by data from the copy source area and all data being read from the target area before the initiation of the copy operation will not reflect the results of the copy operation.

[0073] Another aspect of maintaining point in time copy semantics is ensuring that no data written to the copy source area after the initiation of the copy will appear in the copy target area, no data written to the copy target area after the initiation of the copy will be overwritten by data from the copy source area and all data read from the copy target area after the initiation of the copy will reflect the completion of the copy operation and will not result in reading the data previously stored in the target area. Figures 14 and 15 depict the steps performed by data file management system for snapshot copy operations 106 within controller 104 to achieve this second aspect of maintaining point in time copy semantics.

[0074] Figure 14 illustrates in flow diagram form the operational steps taken by controller 104 when screening data file read requests received from processor 101 by data storage subsystem 100. At step 1401, controller 104 determines whether the data file read request specifies a Virtual Track Address which is the target of an in-progress snapshot copy operation. If the data file read request is for a virtual track which is the target of an in-progress snapshot copy operation and the update of the Virtual Track Table entry selected by the data file's Virtual Track Address has not yet been performed by the snapshot copy process, processing proceeds with step 1402.

[0075] At step 1402, controller 104 translates the Virtual Track Address of the data file which is the target of the snapshot copy operation to the Virtual Track Address of the data file which is being copied. This translated Virtual Track Address is then used for all subsequent operational steps required to complete the processor 101 read file request. In this way, data storage subsystem 100 transfers the copy source data file to processor 101 in response to the data file read request. Thus, the data file received by processor 101 in response to a data file read request that is received by data storage subsystem 100 while a snapshot copy operation is in progress is the same as the data file that would have been received by processor 101 in response to the data file read request had the snapshot copy operation been performed instantly.

[0076] Alternatively, if controller 104 determines at step 1401 that either the data file read request is not for a copy target data file or the snapshot copy operation has already copied the Track Number describing the source data file to the Virtual Track Table entry which describes the target data file, processing proceeds with step 1403. The step 1403, processor 104 uses the Virtual Track Address specified by processor 101 in the read file request for all subsequent operational steps re-

quired to complete the read file request. These subsequent operational steps are the usual steps performed by a dynamically mapped virtual storage subsystem.

[0077] Figure 15 illustrates in flow diagram form the operational steps taken by controller 104 when screening virtual track write requests which are generated by data file management system for snapshot copy operations 106 for the purpose of writing new or modified virtual tracks to backend data storage devices 105. At step 5, controller 104 determines whether the virtual track is the target of an in-progress snapshot copy operation and the Virtual Track Table entry selected by the Virtual Track Address of the Virtual track being written is within a Virtual Track Table Page which the snapshot copy operation has not yet written to backend data storage devices 105. If this is so, processing proceeds with step 1502. At step 1502, controller 104 delays the writing of the virtual track until the in-progress snapshot copy operation has written the Virtual Track Table Page 10 that contains the Virtual Track Table entry selected by the Virtual Track Address assigned by processor 101 when it wrote or modified the virtual track. This prevents any modified data written by processor 101 to the data file contained in the virtual track subsequent to the initiation of the snapshot copy operation from being overwritten by the data file contained in the source virtual track that is being copied.

[0078] If controller 104 determines at step 1501 that the writing of the virtual track did not have to be delayed due to the track being the target of an in-progress snapshot copy, processing proceeds with step 1503. At step 1503, controller 104 determines whether the virtual track is the source of an in-progress snapshot copy operation and the Virtual Track Table entry selected by the Virtual Track Address of the virtual track to which the current virtual track is being copied is within a Virtual Track Table Page which the snapshot copy operation has not yet written to backend data storage devices 105. If this is so, processing proceeds with step 1504. At step 40, controller 104 delays the writing of the virtual track until the in-progress snapshot copy operation has written the Virtual Track Table Page that contains the Virtual Track Table entry selected by the Virtual Track Address of the virtual track to which the current virtual track is being copied. This prevents any modified data written by processor 101 to the data file contained in the virtual track subsequent to the initiation of the snapshot copy operation from being copied to the data file which is the target of the snapshot copy operation.

[0079] If controller 104 determines at step 1503 that the writing of the virtual track did not have to be delayed due to the track being the source of an in-progress snapshot copy, processing proceeds with step 1505. At step 55, controller 104 allows the write of the virtual track instance to proceed without delay.

**Summary**

[0080] Thus, the present data file storage management system for snapshot copy operations 106 both efficiently ensures the reliability of the mapping table data and also performs an incremental copy process which preserves the point in time copy semantics to ensure copy data file correspondence to the original data file. This system maintains a two level mapping table which enables the data files to be copied using the snapshot copy process 107 and only having to update a single corresponding mapping table entry when the physical location of the data file is changed. In addition, the synchronization of the snapshot copy operation with the reading and writing of data to the original and copy data files is maintained by detecting accesses to the original data file or the copy file during the time that the snapshot copy process 107 is being executed and the mapping table is being updated.

**Claims**

1. A data file storage management system for use in a dynamically mapped virtual data storage subsystem (100) to maintain data file copy integrity in snapshot copy operations, wherein said dynamically mapped virtual data storage subsystem (100) comprises a set of backend data storage devices (105) for the storage of data files, each having assigned thereto a virtual track address, received from at least one processor (101) which is connected to said dynamically mapped virtual data storage subsystem (100), said data file storage management system comprising:

means for assigning an immutable identification to said data file received from said processor (101);  
 means for storing data in a first mapping table to associate said virtual track address assigned to said received data file with said assigned immutable identification;  
 means for storing data in a second mapping table to associate said immutable identification to a logical address indicative of a physical storage location on one of said backend data storage devices (105) for the storage of said received data file; and  
 means for writing said received data file at said logical address.

2. The system of claim 1 further comprising:

means for storing a copy of said first mapping table on said backend data storage devices.

3. The system of claim 2 further comprising:

means, responsive to loss of said first map-

ping table, for retrieving said copy of said first mapping table from said backend data storage devices.

4. The system of claim 1 wherein said means for storing data in a first mapping table comprises:

means for storing data indicative of said virtual track address assigned to said received data file by said processor; and  
 means for storing said assigned immutable identification in a manner to correspond said assigned immutable identification to said virtual track address.

5. The system of claim 4 wherein said means for storing data in a second mapping table comprises:

means for storing said assigned immutable identification;  
 means for storing data indicative of a number of virtual track addresses corresponding to said assigned immutable identification; and  
 means for storing data indicative of said logical address assigned to said stored data file in a manner to correspond said assigned immutable identification to said logical address.

6. The system of claim 5 further comprising:

means, responsive to relocation of said stored data file to another logical address in said backend data storage devices, for updating said second mapping table to reflect location of said stored data file at said another logical address, exclusive of updating said first mapping table.

7. The system of claim 5 further comprising:

means, responsive to receipt of a request from said processor to copy said received data file to a second virtual track address, for assigning the immutable identification associated with said received data file to said second virtual track address; and  
 means for activating said means for storing data in a second mapping table to store data in said second mapping table to associate said second immutable identification to said logical address indicative of said physical storage location on a one of said backend data storage devices which stores said received data file.

8. The system of claim 7 further comprising:

means, responsive to said means for storing data in a first mapping table storing said data in said first mapping table to associate said second virtual track address with said immutable identification, for incrementing said data indicative of a number of virtual track addresses corresponding to said as-

signed immutable identification.

9. The system of claim 8 further comprising:  
 means, responsive to receipt of a request from said processor to modify said received data file during processing of said request from said processor to copy said received data file to a second virtual track address, for delaying storing of the modified form of said received data file on said backend data storage devices. 5

10. The system of claim 7 further comprising:  
 means, responsive to means for storing data in a first mapping table storing said data in said first mapping table to associate said second virtual track address with said immutable identification, for storing said first mapping table on said backend data storage devices. 15

11. The system of claim 7 further comprising:  
 means, responsive to means for storing data in a first mapping table storing said data in said first mapping table to associate said second virtual track address with said immutable identification, for storing the portion of said first mapping table that contains said stored data on said backend data storage devices. 20

12. The system of claim 11 further comprising:  
 means, responsive to receipt of a request from said processor to modify said received data file during processing of said request from said processor to copy said received data file to a second virtual track address, for delaying the storing of the modified form of said received data file on said backend data storage devices until after said modified portion of said first mapping table has been stored on said backend data storage devices. 30

13. The system of claim 1 wherein said means for assigning comprises:  
 means for maintaining a list of unused immutable identifications;  
 means, responsive to receipt of a data file having a virtual track address from said processor, for selecting a one of said unused immutable identifications; and  
 means for associating said selected unused immutable identification with said received data file. 40 50

14. A method of data file storage management for use in a dynamically mapped virtual data storage subsystem (100) to maintain data file copy integrity in snapshot copy operations, wherein said dynamically mapped virtual data storage subsystem (100) comprises a set of backend data storage devices 55

(105) for the storage of data files, each having assigned thereto a virtual track address, received from at least one processor (101) which is connected to said dynamically mapped virtual data storage subsystem (100), said data file storage management method comprising the steps of:  
 assigning an immutable identification to said data file received from said processor (101), storing data in a first mapping table to associate said virtual track address assigned to said received data file with said assigned immutable identification;  
 storing data in a second mapping table to associate said immutable identification to a logical address indicative of a physical storage location on one of said backend data storage devices (105) for the storage of said received data file; and  
 writing said received data file at said logical address. 10

15. The method of claim 14 further comprising the step of:  
 storing a copy of said first mapping table on said backend data storage devices. 25

16. The method of claim 15 further comprising the step of:  
 retrieving, in response to loss of said first mapping table, said copy of said first mapping table from said backend data storage devices. 35

17. The method of claim 14 wherein said step of storing data in a first mapping table comprises:  
 storing data indicative of said virtual track address assigned to said received data file by said processor; and  
 storing said assigned immutable identification in a manner to correspond said assigned immutable identification to said virtual track address. 45

18. The method of claim 17 wherein said step of storing data in a second mapping table comprises:  
 storing said assigned immutable identification; storing data indicative of a number of virtual track addresses corresponding to said assigned immutable identification; and  
 storing data indicative of said logical address assigned to said stored data file in a manner to correspond said assigned immutable identification to said logical address. 55

19. The method of claim 18 further comprising the step of:

updating, in response to relocation of said stored data file to another logical address in said backend data storage devices 105, said second mapping table to reflect location of said stored data file at said another logical address, exclusive of updating said first mapping table.

20. The method of claim 18 further comprising the steps of:

assigning, in response to receipt of a request from said processor to copy said received data file to a second virtual track address, the immutable identification associated with said received data file to said second virtual track address; and activating said step of storing data in a second mapping table to store data in said second mapping table to associate said second immutable identification to said logical address indicative of said physical storage location on a one of said backend data storage devices which stores said received data file.

21. The method of claim 20 further comprising the step of:

incrementing, in response to said means for storing data in a first mapping table storing said data in said first mapping table to associate said second virtual track address with said immutable identification, said data indicative of a number of virtual track addresses corresponding to said assigned immutable identification.

22. The method of claim 21 further comprising the step of:

delaying, in response to receipt of a request from said processor to modify said received data file during processing of said request from said processor to copy said received data file to a second virtual track address, storing of the modified form of said received data file on said backend data storage devices.

23. The method of claim 22 further comprising the step of:

storing, in response to storing data in a first mapping table storing said data in said first mapping table to associate said second virtual track address with said immutable identification, said first mapping table on said backend data storage devices.

24. The method of claim 22 further comprising the step of:

storing, in response to storing data in a first mapping table storing said data in said first mapping table to associate said second virtual track address with said immutable identification, the portion of

5 said first mapping table that contains said stored data on said backend data storage devices.

25. The method of claim 11 further comprising the step of:

delaying, in response to receipt of a request from said processor to modify said received data file during processing of said request from said processor to copy said received data file to a second virtual track address, the storing of the modified form of said received data file on said backend data storage devices until after said modified portion of said first mapping table has been stored on said backend data storage devices.

10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425 430 435 440 445 450 455 460 465 470 475 480 485 490 495 500 505 510 515 520 525 530 535 540 545 550 555 560 565 570 575 580 585 590 595 600 605 610 615 620 625 630 635 640 645 650 655 660 665 670 675 680 685 690 695 700 705 710 715 720 725 730 735 740 745 750 755 760 765 770 775 780 785 790 795 800 805 810 815 820 825 830 835 840 845 850 855 860 865 870 875 880 885 890 895 900 905 910 915 920 925 930 935 940 945 950 955 960 965 970 975 980 985 990 995 1000 1005 1010 1015 1020 1025 1030 1035 1040 1045 1050 1055 1060 1065 1070 1075 1080 1085 1090 1095 1100 1105 1110 1115 1120 1125 1130 1135 1140 1145 1150 1155 1160 1165 1170 1175 1180 1185 1190 1195 1200 1205 1210 1215 1220 1225 1230 1235 1240 1245 1250 1255 1260 1265 1270 1275 1280 1285 1290 1295 1300 1305 1310 1315 1320 1325 1330 1335 1340 1345 1350 1355 1360 1365 1370 1375 1380 1385 1390 1395 1400 1405 1410 1415 1420 1425 1430 1435 1440 1445 1450 1455 1460 1465 1470 1475 1480 1485 1490 1495 1500 1505 1510 1515 1520 1525 1530 1535 1540 1545 1550 1555 1560 1565 1570 1575 1580 1585 1590 1595 1600 1605 1610 1615 1620 1625 1630 1635 1640 1645 1650 1655 1660 1665 1670 1675 1680 1685 1690 1695 1700 1705 1710 1715 1720 1725 1730 1735 1740 1745 1750 1755 1760 1765 1770 1775 1780 1785 1790 1795 1800 1805 1810 1815 1820 1825 1830 1835 1840 1845 1850 1855 1860 1865 1870 1875 1880 1885 1890 1895 1900 1905 1910 1915 1920 1925 1930 1935 1940 1945 1950 1955 1960 1965 1970 1975 1980 1985 1990 1995 2000 2005 2010 2015 2020 2025 2030 2035 2040 2045 2050 2055 2060 2065 2070 2075 2080 2085 2090 2095 2100 2105 2110 2115 2120 2125 2130 2135 2140 2145 2150 2155 2160 2165 2170 2175 2180 2185 2190 2195 2200 2205 2210 2215 2220 2225 2230 2235 2240 2245 2250 2255 2260 2265 2270 2275 2280 2285 2290 2295 2300 2305 2310 2315 2320 2325 2330 2335 2340 2345 2350 2355 2360 2365 2370 2375 2380 2385 2390 2395 2400 2405 2410 2415 2420 2425 2430 2435 2440 2445 2450 2455 2460 2465 2470 2475 2480 2485 2490 2495 2500 2505 2510 2515 2520 2525 2530 2535 2540 2545 2550 2555 2560 2565 2570 2575 2580 2585 2590 2595 2600 2605 2610 2615 2620 2625 2630 2635 2640 2645 2650 2655 2660 2665 2670 2675 2680 2685 2690 2695 2700 2705 2710 2715 2720 2725 2730 2735 2740 2745 2750 2755 2760 2765 2770 2775 2780 2785 2790 2795 2800 2805 2810 2815 2820 2825 2830 2835 2840 2845 2850 2855 2860 2865 2870 2875 2880 2885 2890 2895 2900 2905 2910 2915 2920 2925 2930 2935 2940 2945 2950 2955 2960 2965 2970 2975 2980 2985 2990 2995 3000 3005 3010 3015 3020 3025 3030 3035 3040 3045 3050 3055 3060 3065 3070 3075 3080 3085 3090 3095 3100 3105 3110 3115 3120 3125 3130 3135 3140 3145 3150 3155 3160 3165 3170 3175 3180 3185 3190 3195 3200 3205 3210 3215 3220 3225 3230 3235 3240 3245 3250 3255 3260 3265 3270 3275 3280 3285 3290 3295 3300 3305 3310 3315 3320 3325 3330 3335 3340 3345 3350 3355 3360 3365 3370 3375 3380 3385 3390 3395 3400 3405 3410 3415 3420 3425 3430 3435 3440 3445 3450 3455 3460 3465 3470 3475 3480 3485 3490 3495 3500 3505 3510 3515 3520 3525 3530 3535 3540 3545 3550 3555 3560 3565 3570 3575 3580 3585 3590 3595 3600 3605 3610 3615 3620 3625 3630 3635 3640 3645 3650 3655 3660 3665 3670 3675 3680 3685 3690 3695 3700 3705 3710 3715 3720 3725 3730 3735 3740 3745 3750 3755 3760 3765 3770 3775 3780 3785 3790 3795 3800 3805 3810 3815 3820 3825 3830 3835 3840 3845 3850 3855 3860 3865 3870 3875 3880 3885 3890 3895 3900 3905 3910 3915 3920 3925 3930 3935 3940 3945 3950 3955 3960 3965 3970 3975 3980 3985 3990 3995 4000 4005 4010 4015 4020 4025 4030 4035 4040 4045 4050 4055 4060 4065 4070 4075 4080 4085 4090 4095 4100 4105 4110 4115 4120 4125 4130 4135 4140 4145 4150 4155 4160 4165 4170 4175 4180 4185 4190 4195 4200 4205 4210 4215 4220 4225 4230 4235 4240 4245 4250 4255 4260 4265 4270 4275 4280 4285 4290 4295 4300 4305 4310 4315 4320 4325 4330 4335 4340 4345 4350 4355 4360 4365 4370 4375 4380 4385 4390 4395 4400 4405 4410 4415 4420 4425 4430 4435 4440 4445 4450 4455 4460 4465 4470 4475 4480 4485 4490 4495 4500 4505 4510 4515 4520 4525 4530 4535 4540 4545 4550 4555 4560 4565 4570 4575 4580 4585 4590 4595 4600 4605 4610 4615 4620 4625 4630 4635 4640 4645 4650 4655 4660 4665 4670 4675 4680 4685 4690 4695 4700 4705 4710 4715 4720 4725 4730 4735 4740 4745 4750 4755 4760 4765 4770 4775 4780 4785 4790 4795 4800 4805 4810 4815 4820 4825 4830 4835 4840 4845 4850 4855 4860 4865 4870 4875 4880 4885 4890 4895 4900 4905 4910 4915 4920 4925 4930 4935 4940 4945 4950 4955 4960 4965 4970 4975 4980 4985 4990 4995 5000 5005 5010 5015 5020 5025 5030 5035 5040 5045 5050 5055 5060 5065 5070 5075 5080 5085 5090 5095 5100 5105 5110 5115 5120 5125 5130 5135 5140 5145 5150 5155 5160 5165 5170 5175 5180 5185 5190 5195 5200 5205 5210 5215 5220 5225 5230 5235 5240 5245 5250 5255 5260 5265 5270 5275 5280 5285 5290 5295 5300 5305 5310 5315 5320 5325 5330 5335 5340 5345 5350 5355 5360 5365 5370 5375 5380 5385 5390 5395 5400 5405 5410 5415 5420 5425 5430 5435 5440 5445 5450 5455 5460 5465 5470 5475 5480 5485 5490 5495 5500 5505 5510 5515 5520 5525 5530 5535 5540 5545 5550 5555 5560 5565 5570 5575 5580 5585 5590 5595 5600 5605 5610 5615 5620 5625 5630 5635 5640 5645 5650 5655 5660 5665 5670 5675 5680 5685 5690 5695 5700 5705 5710 5715 5720 5725 5730 5735 5740 5745 5750 5755 5760 5765 5770 5775 5780 5785 5790 5795 5800 5805 5810 5815 5820 5825 5830 5835 5840 5845 5850 5855 5860 5865 5870 5875 5880 5885 5890 5895 5900 5905 5910 5915 5920 5925 5930 5935 5940 5945 5950 5955 5960 5965 5970 5975 5980 5985 5990 5995 6000 6005 6010 6015 6020 6025 6030 6035 6040 6045 6050 6055 6060 6065 6070 6075 6080 6085 6090 6095 6100 6105 6110 6115 6120 6125 6130 6135 6140 6145 6150 6155 6160 6165 6170 6175 6180 6185 6190 6195 6200 6205 6210 6215 6220 6225 6230 6235 6240 6245 6250 6255 6260 6265 6270 6275 6280 6285 6290 6295 6300 6305 6310 6315 6320 6325 6330 6335 6340 6345 6350 6355 6360 6365 6370 6375 6380 6385 6390 6395 6400 6405 6410 6415 6420 6425 6430 6435 6440 6445 6450 6455 6460 6465 6470 6475 6480 6485 6490 6495 6500 6505 6510 6515 6520 6525 6530 6535 6540 6545 6550 6555 6560 6565 6570 6575 6580 6585 6590 6595 6600 6605 6610 6615 6620 6625 6630 6635 6640 6645 6650 6655 6660 6665 6670 6675 6680 6685 6690 6695 6700 6705 6710 6715 6720 6725 6730 6735 6740 6745 6750 6755 6760 6765 6770 6775 6780 6785 6790 6795 6800 6805 6810 6815 6820 6825 6830 6835 6840 6845 6850 6855 6860 6865 6870 6875 6880 6885 6890 6895 6900 6905 6910 6915 6920 6925 6930 6935 6940 6945 6950 6955 6960 6965 6970 6975 6980 6985 6990 6995 7000 7005 7010 7015 7020 7025 7030 7035 7040 7045 7050 7055 7060 7065 7070 7075 7080 7085 7090 7095 7100 7105 7110 7115 7120 7125 7130 7135 7140 7145 7150 7155 7160 7165 7170 7175 7180 7185 7190 7195 7200 7205 7210 7215 7220 7225 7230 7235 7240 7245 7250 7255 7260 7265 7270 7275 7280 7285 7290 7295 7300 7305 7310 7315 7320 7325 7330 7335 7340 7345 7350 7355 7360 7365 7370 7375 7380 7385 7390 7395 7400 7405 7410 7415 7420 7425 7430 7435 7440 7445 7450 7455 7460 7465 7470 7475 7480 7485 7490 7495 7500 7505 7510 7515 7520 7525 7530 7535 7540 7545 7550 7555 7560 7565 7570 7575 7580 7585 7590 7595 7600 7605 7610 7615 7620 7625 7630 7635 7640 7645 7650 7655 7660 7665 7670 7675 7680 7685 7690 7695 7700 7705 7710 7715 7720 7725 7730 7735 7740 7745 7750 7755 7760 7765 7770 7775 7780 7785 7790 7795 7800 7805 7810 7815 7820 7825 7830 7835 7840 7845 7850 7855 7860 7865 7870 7875 7880 7885 7890 7895 7900 7905 7910 7915 7920 7925 7930 7935 7940 7945 7950 7955 7960 7965 7970 7975 7980 7985 7990 7995 8000 8005 8010 8015 8020 8025 8030 8035 8040 8045 8050 8055 8060 8065 8070 8075 8080 8085 8090 8095 8100 8105 8110 8115 8120 8125 8130 8135 8140 8145 8150 8155 8160 8165 8170 8175 8180 8185 8190 8195 8200 8205 8210 8215 8220 8225 8230 8235 8240 8245 8250 8255 8260 8265 8270 8275 8280 8285 8290 8295 8300 8305 8310 8315 8320 8325 8330 8335 8340 8345 8350 8355 8360 8365 8370 8375 8380 8385 8390 8395 8400 8405 8410 8415 8420 8425 8430 8435 8440 8445 8450 8455 8460 8465 8470 8475 8480 8485 8490 8495 8500 8505 8510 8515 8520 8525 8530 8535 8540 8545 8550 8555 8560 8565 8570 8575 8580 8585 8590 8595 8600 8605 8610 8615 8620 8625 8630 8635 8640 8645 8650 8655 8660 8665 8670 8675 8680 8685 8690 8695 8700 8705 8710 8715 8720 8725 8730 8735 8740 8745 8750 8755 8760 8765 8770 8775 8780 8785 8790 8795 8800 8805 8810 8815 8820 8825 8830 8835 8840 8845 8850 8855 8860 8865 8870 8875 8880 8885 8890 8895 8900 8905 8910 8915 8920 8925 8930 8935 8940 8945 8950 8955 8960 8965 8970 8975 8980 8985 8990 8995 9000 9005 9010 9015 9020 9025 9030 9035 9040 9045 9050 9055 9060 9065 9070 9075 9080 9085 9090 9095 9100 9105 9110 9115 9120 9125 9130 9135 9140 9145 9150 9155 9160 9165 9170 9175 9180 9185 9190 9195 9200 9205 9210 9215 9220 9225 9230 9235 9240 9245 9250 9255 9260 9265 9270 9275 9280 9285 9290 9295 9300 9305 9310 9315 9320 9325 9330 9335 9340 9345 9350 9355 9360 9365 9370 9375 9380 9385 9390 9395 9400 9405 9410 9415 9420 9425 9430 9435 9440 9445 9450 9455 9460 9465 9470 9475 9480 9485 9490 9495 9500 9505 9510 9515 9520 9525 9530 9535 9540 9545 9550 9555 9560 9565 9570 9575 9580 9585 9590 9595 9600 9605 9610 9615 9620 9625 9630 9635 9640 9645 9650 9655 9660 9665 9670 9675 9680 9685 9690 9695 9700 9705 9710 9715 9720 9725 9730 9735 9740 9745 9750 9755 9760 9765 9770 9775 9780 9785 9790 9795 9800 9805 9810 9815 9820 9825 9830 9835 9840 9845 9850 9855 9860 9865 9870 9875 9880 9885 9890 9895 9900 9905 9910 9915 9920 9925 9930 9935 9940 9945 9950 9955 9960 9965 9970 9975 9980 9985 9990 9995 9999

chervorrichtungen (105) für die Speicherung der empfangenen Datendatei, zuzuordnen; und eine Einrichtung zum Schreiben der empfangenen Datendatei an der logischen Adresse. 5

2. System nach Anspruch 1, das weiterhin aufweist: eine Einrichtung zum Speichern einer Kopie der ersten Auflistungstabelle an den Back-End-Datenspeichervorrichtungen. 10

3. System nach Anspruch 2, das weiterhin aufweist: eine Einrichtung, die auf einen Verlust der ersten Auflistungstabelle anspricht, um die Kopie der ersten Auflistungstabelle von den Back-End-Datenspeichervorrichtungen aufzusuchen. 15

4. System nach Anspruch 1, wobei die Einrichtung zum Speichern von Daten in einer ersten Auflistungstabelle aufweist: 20

    eine Einrichtung zum Speichern von Daten, die für die virtuelle Spur-Adresse, zugeordnet zu der empfangenen Datendatei durch den Prozessor, indikativ sind; und eine Einrichtung zum Speichern der zugeordneten, unveränderbaren Identifikation in einer Art und Weise, um der zugeordneten, unveränderbaren Identifikation zu der virtuellen Spur-Adresse zu entsprechen. 25

5. System nach Anspruch 4, wobei die Einrichtung zum Speichern von Daten in einer zweiten Auflistungstabelle aufweist: 30

    eine Einrichtung zum Speichern der zugeordneten, unveränderbaren Identifikation; eine Einrichtung zum Speichern von Daten, die für eine Anzahl von virtuellen Spur-Adressen entsprechend zu der zugeordneten, unveränderbaren Identifikation indikativ sind; und eine Einrichtung zum Speichern von Daten, die für die logische Adresse indikativ sind, die zu der gespeicherten Datendatei zugeordnet sind, in einer Art und Weise, um der zugeordneten, unveränderbaren Identifikation zu der logischen Adresse zu entsprechen. 35

6. System nach Anspruch 5, das weiterhin aufweist: eine Einrichtung, die auf eine erneute Zuordnung der gespeicherten Datendatei zu einer anderen, logischen Adresse in den Back-End-Datenspeichervorrichtungen anspricht, zum Aktualisieren der zweiten Auflistungstabelle, um eine Stelle der gespeicherten Datendatei an der anderen, logischen Adresse, ohne eine Aktualisierung der ersten Auflistungstabelle, zu reflektieren. 40

7. System nach Anspruch 5, das weiterhin aufweist: eine Einrichtung, die auf einen Empfang einer Anforderung von dem Prozessor anspricht, um die empfangene Datendatei zu einer zweiten, virtuellen Spur-Adresse zu kopieren, zum Zuordnen der unveränderbaren Identifikation, die der empfangenen Datendatei zugeordnet ist, zu der zweiten, virtuellen Spur-Adresse; und eine Einrichtung zum Aktivieren der Einrichtung zum Speichern von Daten in einer zweiten Auflistungstabelle, um Daten in der zweiten Auflistungstabelle zu speichern, um die zweite, unveränderbare Identifikation zu der logischen Adresse, indikativ für die physikalische Speicherstelle an einer der Back-End-Datenspeichervorrichtungen, die die empfangene Datendatei speichert, zuzuordnen. 45

8. System nach Anspruch 7, das weiterhin aufweist: eine Einrichtung, die auf die Einrichtung zum Speichern von Daten in einer ersten Auflistungstabelle, die die Daten in der ersten Auflistungstabelle speichert, anspricht, um die zweite, virtuelle Spur-Adresse der unveränderbaren Identifikation zuzuordnen, zum Erhöhen der Daten, die für eine Zahl virtueller Spur-Adressen, entsprechend zu der zugeordneten, unveränderbaren Identifikation, indikativ sind. 50

9. System nach Anspruch 8, das weiterhin aufweist: eine Einrichtung, die auf einen Empfang einer Anforderung von dem Prozessor anspricht, um die empfangene Datendatei während einer Verarbeitung der Anforderung von dem Prozessor, um die empfangene Datendatei zu einer zweiten, virtuellen Spur-Adresse zu kopieren, zum Verzögern einer Speicherung der modifizierten Form der empfangenen Datendatei an den Back-End-Datenspeichervorrichtungen, zu modifizieren. 55

10. System nach Anspruch 7, das weiterhin aufweist: eine Einrichtung, die auf eine Einrichtung zum Speichern von Daten in einer ersten Auflistungstabelle, die die Daten in der ersten Auflistungstabelle speichert, anspricht, um die zweite, virtuelle Spur-Adresse mit der unveränderbaren Identifikation, zum Speichern der ersten Auflistungstabelle an den Back-End-Datenspeichervorrichtungen, zuzuordnen.

11. System nach Anspruch 7, das weiterhin aufweist: eine Einrichtung, die auf eine Einrichtung zum Speichern von Daten in einer ersten Auflistungstabelle, die die Daten in der ersten Auflistungstabelle speichert, anspricht, um die zweite, virtuelle Spur-Adresse der unveränderbaren Identifikation, zum Speichern des Teils der ersten Auflistungstabelle,

der die gespeicherten Daten, an den Back-End-Datenspeichervorrichtungen enthält, zuzuordnen.

12. System nach Anspruch 11, das weiterhin aufweist: eine Einrichtung, die auf einen Empfang einer Anforderung von dem Prozessor anspricht, um die empfangene Datendatei während einer Verarbeitung der Anforderung von dem Prozessor zu modifizieren, um die empfangene Datendatei zu einer zweiten, virtuellen Spur-Adresse zu kopieren, zum Verzögern der Speicherung in der modifizierten Form der empfangenen Datendatei an den Back-End-Datenspeichervorrichtungen, bis dann, nachdem der modifizierte Teil der ersten Auflistungstabelle an den Back-End-Datenspeichervorrichtungen gespeichert worden ist. 5

13. System nach Anspruch 1, wobei die Einrichtung zum Zuordnen aufweist: 10

    eine Einrichtung zum Beibehalten einer Liste von nicht verwendeten, unveränderbaren Identifikationen, 20

    eine Einrichtung, die auf einen Empfang einer Datendatei, die eine virtuelle Spur-Adresse von dem Prozessor besitzt, zum Auswählen einer der nicht benutzten, unveränderbaren Identifikation, anspricht, und 25

    eine Einrichtung zum Zuordnen der ausgewählten, nicht verwendeten, unveränderbaren Identifikation zu der empfangenen Datendatei. 30

14. Verfahren eines Datendateispeicher-Managements zur Verwendung in einem Speicheruntersystem (100) für dynamisch aufgelistete, virtuelle Daten, um eine Datendatei-Kopierintegrität in Schnappschuß-Kopieroperationen beizubehalten, wobei das Speicheruntersystem (100) für dynamisch aufgelistete, virtuelle Daten einen Satz von Back-End-Datenspeichervorrichtungen (105) für die Speicherung von Datendateien aufweist, wobei jede dazu zugeordnet eine virtuelle Spur-Adresse besitzt, empfangen von mindestens einem Prozessor (101), der mit dem Speicheruntersystem (100) für dynamisch aufgelistete, virtuelle Daten verbunden ist, wobei das Datendateispeicher-Managementverfahren die Schritte aufweist: 40

    Zuordnen einer unveränderbaren Identifikation zu der Datendatei, die von dem Prozessor (101) empfangen ist; 50

    Speichern von Daten in einer ersten Auflistungstabelle, um die virtuelle Spur-Adresse, zugeordnet zu der empfangenen Datendatei, zu der zugeordneten, unveränderbaren Identifikation zuzuordnen; 55

    Speichern von Daten in einer zweiten Auflistungstabelle, um die unveränderbare Identifi- kation einer logischen Adresse zuzuordnen, die für eine physikalische Speicherstelle an einer der Back-End-Datenspeichervorrichtungen (105) indikativ ist, für die Speicherung der empfangenen Datendatei; und Schreiben der empfangenen Datendatei an der logischen Adresse. 15

15. Verfahren nach Anspruch 14, das weiterhin den Schritt aufweist: 20

    Speichern einer Kopie der ersten Auflistungstabelle an den Back-End-Datenspeichervorrichtungen. 16

16. Verfahren nach Anspruch 15, das weiterhin den Schritt aufweist: 25

    Aufsuchen, auf einen Verlust der ersten Auflistungstabelle hin, der Kopie der ersten Auflistungstabelle von den Back-End-Datenspeichervorrichtungen. 17

17. Verfahren nach Anspruch 14, wobei der Schritt eines Speicherns von Daten in einer ersten Auflistungstabelle aufweist: 30

    Speichern von Daten, die für die virtuelle Spur-Adresse indikativ sind, zugeordnet zu der empfangenen Datendatei durch den Prozessor; und 35

    Speichern der zugeordneten, unveränderbaren Identifikation in einer Art und Weise, um der zugeordneten, unveränderbaren Identifikation, zu der virtuellen Spur-Adresse, zu entsprechen. 18

18. Verfahren nach Anspruch 17, wobei der Schritt eines Speicherns von Daten in einer zweiten Auflistungstabelle aufweist: 40

    Speichern der zugeordneten, unveränderbaren Identifikation; 45

    Speichern von Daten, die für eine Zahl von virtuellen Spur-Adressen entsprechend zu der zugeordneten, unveränderbaren Identifikation indikativ sind; und 50

    Speichern von Daten, die für die logische Adresse, zugeordnet zu der gespeicherten Datendatei, in einer Art und Weise, um der zugeordneten, unveränderbaren Identifikation zu entsprechen, zu der logischen Adresse, indikativ sind. 19

19. Verfahren nach Anspruch 18, das weiterhin den Schritt aufweist: 55

    Aktualisieren, in Abhängigkeit einer erneuten Zuordnung der gespeicherten Datendatei zu einer anderen logischen Adresse in den Back-End-Datenspeichervorrichtungen (105), der zweiten Auflistungstabelle, um eine Stelle der gespeicherten

Datendatei an der anderen, logischen Adresse, ohne ein Aktualisieren der ersten Auflistungstabelle, zu reflektieren.

20. Verfahren nach Anspruch 18, das weiterhin die Schritte aufweist: 5

Zuordnen, in Abhängigkeit von dem Empfang einer Anforderung von dem Prozessor, um die empfangene Datendatei zu einer zweiten, virtuellen Adresse zu kopieren, der unveränderbaren Identifikation, die der empfangenen Datendatei zu der zweiten, virtuellen Spur-Adresse zugeordnet ist; und 10

Aktivieren des Schritts eines Speicherns von Daten in einer zweiten Auflistungstabelle, um Daten in der zweiten Auflistungstabelle zu speichern, um die zweite, unveränderbare Identifikation zu der logischen Adresse zuzuordnen, die für die physikalische Speicherstelle an einer der Back-End-Datenspeichervorrichtungen, die die empfangene Datendatei speichert, indikativ ist. 15

21. Verfahren nach Anspruch 20, das weiterhin den Schritt aufweist: 20

Erhöhen, in Abhängigkeit von der ersten Einrichtung zum Speichern von Daten in einer ersten Auflistungstabelle, die die Daten in der ersten Auflistungstabelle speichert, um die zweite, virtuelle Spur-Adresse zu der unveränderbaren Identifikation zuzuordnen, der Daten, die für eine Zahl von virtuellen Spur-Adressen entsprechend zu der zugeordneten, unveränderbaren Identifikation Indikativ sind. 25

22. Verfahren nach Anspruch 21, das weiterhin den Schritt aufweist: 30

Verzögern, in Abhängigkeit eines Empfangs einer Anforderung von dem Prozessor, um die empfangene Datendatei während eines Verarbeitens der Anforderung von dem Prozessor zu modifizieren, um die empfangene Datendatei zu einer zweiten, virtuellen Spur-Adresse zu kopieren, eines Speicherns der modifizierten Form der empfangenen Datendatei an den Back-End-Datenspeichervorrichtungen. 35

23. Verfahren nach Anspruch 22, das weiterhin den Schritt aufweist: 40

Speichern, in Abhängigkeit eines Speicherns von Daten in einer ersten Auflistungstabelle, die die Daten in der ersten Auflistungstabelle speichert, um die zweite, virtuelle Spur-Adresse der unveränderbaren Identifikation zuzuordnen, der ersten Auflistungstabelle an den Back-End-Datenspeichervorrichtungen. 45

24. Verfahren nach Anspruch 22, das weiterhin den Schritt aufweist: 50

Speichern, in Abhängigkeit eines Speicherns von Daten in einer ersten Auflistungstabelle, die die Daten in der ersten Auflistungstabelle speichert, um die zweite, virtuelle Spur-Adresse der unveränderbaren Identifikation zuzuordnen, des Teils der ersten Auflistungstabelle, der die gespeicherten Daten, an den Back-End-Datenspeichervorrichtungen, enthält. 55

25. Verfahren nach Anspruch 11, das weiterhin den Schritt aufweist: 60

Verzögern, in Abhängigkeit eines Empfangs einer Anforderung von dem Prozessor, um die empfangene Datendatei während eines Verarbeitens der Anforderung von dem Prozessor zu modifizieren, um die empfangene Datendatei zu einer zweiten, virtuellen Spur-Adresse zu kopieren, des Speicherns der modifizierten Form der empfangenen Datendatei an den Back-End-Datenspeichervorrichtungen, bis zu einem Zeitpunkt, nachdem der modifizierte Teil der ersten Auflistungstabelle an den Back-End-Datenspeichervorrichtungen gespeichert worden ist. 65

26. Verfahren nach Anspruch 14, wobei der Schritt eines Zuordnens aufweist: 70

Beibehalten einer Liste von nicht verwendeten, unveränderbaren Identifikationen; Auswählen, in Abhängigkeit eines Empfangs einer Datendatei, die eine virtuelle Spur-Adresse besitzt, von dem Prozessor, einer der nicht verwendeten, unveränderbaren Identifikationen; und Zuordnen der ausgewählten, nicht verwendeten, unveränderbaren Identifikation zu der empfangenen Datendatei. 75

#### Revendications

1. Système de gestion de stockage de fichier de données pour une utilisation dans un sous-système de stockage de données virtuelles cartographiées dynamiquement (100) afin de maintenir une intégrité de copie de fichier de données lors d'opérations de copies instantanées, dans lequel ledit sous-système de stockage de données virtuelles cartographiées dynamiquement (100) comprend un jeu de dispositifs de stockage de données arrières (105) pour le stockage de fichiers de données dont chacun se voit assigner une adresse de piste virtuelle, reçue depuis au moins un processeur (101) qui est connecté audit sous-système de stockage de données virtuelles cartographiées dynamiquement (100), ledit système de gestion de stockage de fichiers de données comprenant :

un moyen pour assigner une identification immuable audit fichier de données reçu depuis ledit processeur (101) ;

un moyen pour stocker des données dans une première table de cartographie afin d'associer ladite adresse de piste virtuelle assignée audit fichier de données reçu avec ladite identification immuable assignée ;

un moyen pour stocker des données dans une seconde cartographie afin d'associer ladite identification immuable à une adresse logique qui est indicative d'un emplacement de stockage physique sur l'un desdits dispositifs de stockage de données arrières (105) pour le stockage dudit fichier de données reçu ; et

un moyen pour écrire ledit fichier de données reçu à ladite adresse logique.

2. Système selon la revendication 1, comprenant en outre :

un moyen pour stocker une copie de ladite première table de cartographie sur lesdits dispositifs de stockage de données arrières.

3. Système selon la revendication 2, comprenant en outre :

un moyen, sensible à la perte de ladite première table de cartographie, pour retrouver ladite copie de ladite première table de cartographie à partir desdits dispositifs de stockage de données arrières.

4. Système selon la revendication 1, dans lequel ledit moyen pour stocker des données dans une première table de cartographie comprend :

un moyen pour stocker des données qui sont indicatives de ladite adresse de piste virtuelle assignée audit fichier de données reçu par ledit processeur ; et

un moyen pour stocker ladite identification immuable assignée de manière à faire correspondre ladite identification immuable assignée à ladite adresse de piste virtuelle.

5. Système selon la revendication 4, dans lequel ledit moyen pour stocker des données dans une seconde table de cartographie comprend :

un moyen pour stocker ladite identification immuable assignée ;

un moyen pour stocker des données qui sont indicatives d'un nombre d'adresses de pistes virtuelles correspondant à ladite identification immuable assignée ; et

un moyen pour stocker des données qui sont indicatives de ladite adresse logique assignée audit fichier de données stocké de manière à faire correspondre ladite identification immuable assignée à ladite adresse logique.

6. Système selon la revendication 5, comprenant en outre :

un moyen, sensible à une re-localisation dudit fichier de données stocké au niveau d'une autre adresse logique dans lesdits dispositifs de stockage de données arrières, pour mettre à jour ladite seconde table de cartographie afin de refléter l'emplacement dudit fichier de données stocké au niveau de ladite autre adresse logique, à l'exclusion de la mise à jour de ladite première table de cartographie.

7. Système selon la revendication 5, comprenant en outre :

un moyen, sensible à la réception d'une requête en provenance dudit processeur afin de copier ledit fichier de données reçu à une seconde adresse de piste virtuelle, pour assigner ladite identification immuable qui est associée audit fichier de données reçu à ladite seconde adresse de piste virtuelle ; et

un moyen pour activer ledit moyen pour stocker les données dans une seconde table de cartographie afin de stocker des données dans ladite seconde table de cartographie afin d'associer ladite seconde identification immuable à ladite adresse logique indicative dudit emplacement de stockage physique sur celui desdits dispositifs de stockage de données arrières qui stocke ledit fichier de données reçu.

8. Système selon la revendication 7, comprenant en outre :

un moyen, sensible audit moyen pour stocker des données dans une première table de cartographie qui stocke lesdites données dans ladite première table de cartographie afin d'associer ladite seconde adresse de piste virtuelle avec ladite identification immuable, pour incrémenter lesdites données indicatives d'un nombre d'adresses de pistes virtuelles correspondant à ladite identification immuable assignée.

9. Système selon la revendication 8, comprenant en outre :

un moyen, sensible à la réception d'une requête en provenance dudit processeur pour modifier ledit fichier de données reçu pendant le traitement de ladite requête en provenance dudit processeur afin de copier ledit fichier de données reçu au niveau d'une seconde adresse de piste virtuelle, pour retarder le stockage de la forme modifiée dudit fichier de données reçu sur lesdits dispositifs de stockage de données arrières.

10. Système selon la revendication 7, comprenant en outre :

un moyen, sensible au moyen pour stocker des données dans une première table de cartographie qui stocke lesdites données dans ladite première table de cartographie afin d'associer ladite seconde adresse de piste virtuelle à ladite identification immuable, pour stocker ladite première table de cartographie sur lesdits dispositifs de stockage de données arrières. 10

11. Système selon la revendication 7, comprenant en outre :

un moyen, sensible au moyen pour stocker des données de première table de cartographie qui stocke lesdites données dans ladite première table de cartographie afin d'associer ladite seconde adresse de piste virtuelle à ladite identification immuable, pour stocker la partie de ladite première table de cartographie qui contient lesdites données stockées sur lesdits dispositifs de stockage de données arrières. 15

12. Système selon la revendication 11, comprenant en outre :

un moyen, sensible à la réception d'une requête en provenance dudit processeur pour modifier ledit fichier de données reçu pendant le traitement de ladite requête en provenance dudit processeur afin de copier ledit fichier de données reçu au niveau d'une seconde adresse de piste virtuelle, pour retarder le stockage de la forme modifiée dudit fichier de données reçu sur lesdits dispositifs de stockage de données arrières jusqu'à après que ladite partie modifiée de ladite première table de cartographie a été stockée sur lesdits dispositifs de stockage de données arrières. 20

13. Système selon la revendication 1, dans lequel ledit moyen d'assignation comprend :

un moyen pour maintenir une liste d'identifications immuables inutilisées ;  
un moyen, sensible à la réception d'un fichier de données qui présente une adresse de piste virtuelle en provenance dudit processeur, pour sélectionner l'une desdites identifications immuables inutilisées ; et  
un moyen pour associer ladite identification immuable inutilisée sélectionnée audit fichier de données reçu. 40

14. Procédé de gestion de stockage de fichier de données pour une utilisation dans un sous-système de stockage de données virtuelles cartographiées dynamiquement (100) afin de maintenir une intégrité de copie de fichier de données lors d'opérations de copies instantanées, dans lequel ledit sous-système de stockage de données virtuelles cartographiées dynamiquement (100) comprend un jeu de dispositifs de stockage de données arrières (105) pour le stockage de fichiers de données dont chacun se voit assigner une adresse de piste virtuelle, reçue depuis au moins un processeur (101) qui est connecté audit sous-système de stockage de données virtuelles cartographiées dynamiquement (100), ledit procédé de gestion de stockage de fichier de données comprenant les étapes de :

assignation d'une identification immuable audit fichier de données reçu depuis ledit processeur (101) ;  
stockage de données dans une première table de cartographie afin d'associer ladite adresse de piste virtuelle assignée audit fichier de données reçu à ladite identification immuable assignée ;  
stockage de données dans une seconde table de cartographie afin d'associer ladite identification immuable à une adresse logique qui est indicative d'un emplacement de stockage physique sur l'un desdits dispositifs de stockage de données arrières (105) pour le stockage dudit fichier de données reçu ; et  
écriture dudit fichier de données reçu au niveau de ladite adresse logique. 25

15. Procédé selon la revendication 14, comprenant en outre l'étape de :

stockage d'une copie de ladite première table de cartographie sur lesdits dispositifs de stockage de données arrières. 30

16. Procédé selon la revendication 15, comprenant en outre l'étape de :

recherche, en réponse à la perte de ladite première table de cartographie, de ladite copie de ladite première table de cartographie à partir desdits dispositifs de stockage de données arrières. 35

17. Procédé selon la revendication 14, dans lequel ladite étape de stockage de données dans une première table de cartographie comprend :

le stockage de données qui sont indicatives de ladite adresse de piste virtuelle qui est assignée audit fichier de données reçu par ledit processeur ; et  
le stockage de ladite identification immuable assignée de manière à faire correspondre ladite identification immuable assignée à ladite adresse de piste virtuelle. 45

18. Procédé selon la revendication 17, dans lequel ladite étape de stockage de données dans une seconde table de cartographie comprend :

le stockage de ladite identification immuable assignée ;  
le stockage de données indicatives d'un nombre d'adresses de pistes virtuelles correspondant à ladite identification immuable assignée ; et  
le stockage de données qui sont indicatives de ladite adresse logique qui est assignée audit fichier de données stocké de manière à faire correspondre ladite identification immuable assignée à ladite adresse logique.

19. Procédé selon la revendication 18, comprenant en outre l'étape de :  
mise à jour, en réponse à une re-localisation dudit fichier de données stocké au niveau d'une autre adresse logique dans lesdits dispositifs de stockage de données arrières (105), de ladite seconde table de cartographie afin de refléter l'emplacement dudit fichier de données stocké au niveau de ladite autre adresse logique, à l'exclusion de la mise à jour de ladite première table de cartographie.

20. Procédé selon la revendication 18, comprenant en outre les étapes de :  
assignation, en réponse à la réception d'une requête en provenance dudit processeur pour copier ledit fichier de données reçu au niveau d'une seconde adresse de piste virtuelle, de l'identification immuable qui est associée audit fichier de données reçu à ladite seconde adresse de piste virtuelle ; et  
activation de ladite étape de stockage de données dans une seconde table de cartographie afin de stocker des données dans ladite seconde table de cartographie afin d'associer ladite seconde identification immuable à ladite adresse logique qui est indicative dudit emplacement de stockage physique sur celui desdits dispositifs de stockage de données arrières qui stocke ledit fichier de données reçu.

21. Procédé selon la revendication 20, comprenant en outre l'étape de :  
incrémentation, en réponse audit moyen pour stocker des données d'une première table de cartographie qui stocke lesdites données dans ladite première table de cartographie afin d'associer ladite seconde adresse de piste virtuelle à ladite identification immuable, desdites données qui sont indicatives d'un nombre d'adresses de pistes virtuelles correspondant à ladite identification immuable assignée.

22. Procédé selon la revendication 21, comprenant en outre à l'étape de :  
retardement, en réponse à la réception d'une

5

10 23. Procédé selon la revendication 22, comprenant en outre l'étape de :  
stockage, en réponse au stockage de données d'une première table de cartographie qui stocke lesdites données dans ladite première table de cartographie afin d'associer ladite seconde adresse de piste virtuelle à ladite identification immuable, de ladite première table de cartographie sur lesdits dispositifs de stockage de données arrières.

15

20 24. Procédé selon la revendication 22, comprenant en outre l'étape de :  
stockage, en réponse au stockage de données dans une première table de cartographie qui stocke lesdites données dans ladite première table de cartographie afin d'associer ladite seconde adresse de piste virtuelle à ladite identification immuable, de la partie de ladite première table de cartographie qui contient lesdites données stockées sur lesdits dispositifs de stockage de données arrières.

25

30

35

40

45

50

55

requête en provenance dudit processeur pour modifier ledit fichier de données reçu pendant le traitement de ladite requête en provenance dudit processeur afin de copier ledit fichier de données reçu au niveau d'une seconde adresse de piste virtuelle, du stockage de la forme modifiée dudit fichier de données reçu sur lesdits dispositifs de stockage de données arrières.

26. Procédé selon la revendication 14, dans lequel ladite étape d'assignation comprend :  
le maintien d'une liste d'identifications immuables inutilisées ;  
la sélection, en réponse à la réception d'un fichier de données qui présente une adresse de piste virtuelle en provenance dudit processeur, d'une desdites identifications immuables et inutilisées ; et  
l'association de ladite identification immuable inutilisée sélectionnée avec ledit fichier de don-

nées reçus.

5

10

15

20

25

30

35

40

45

50

55

25

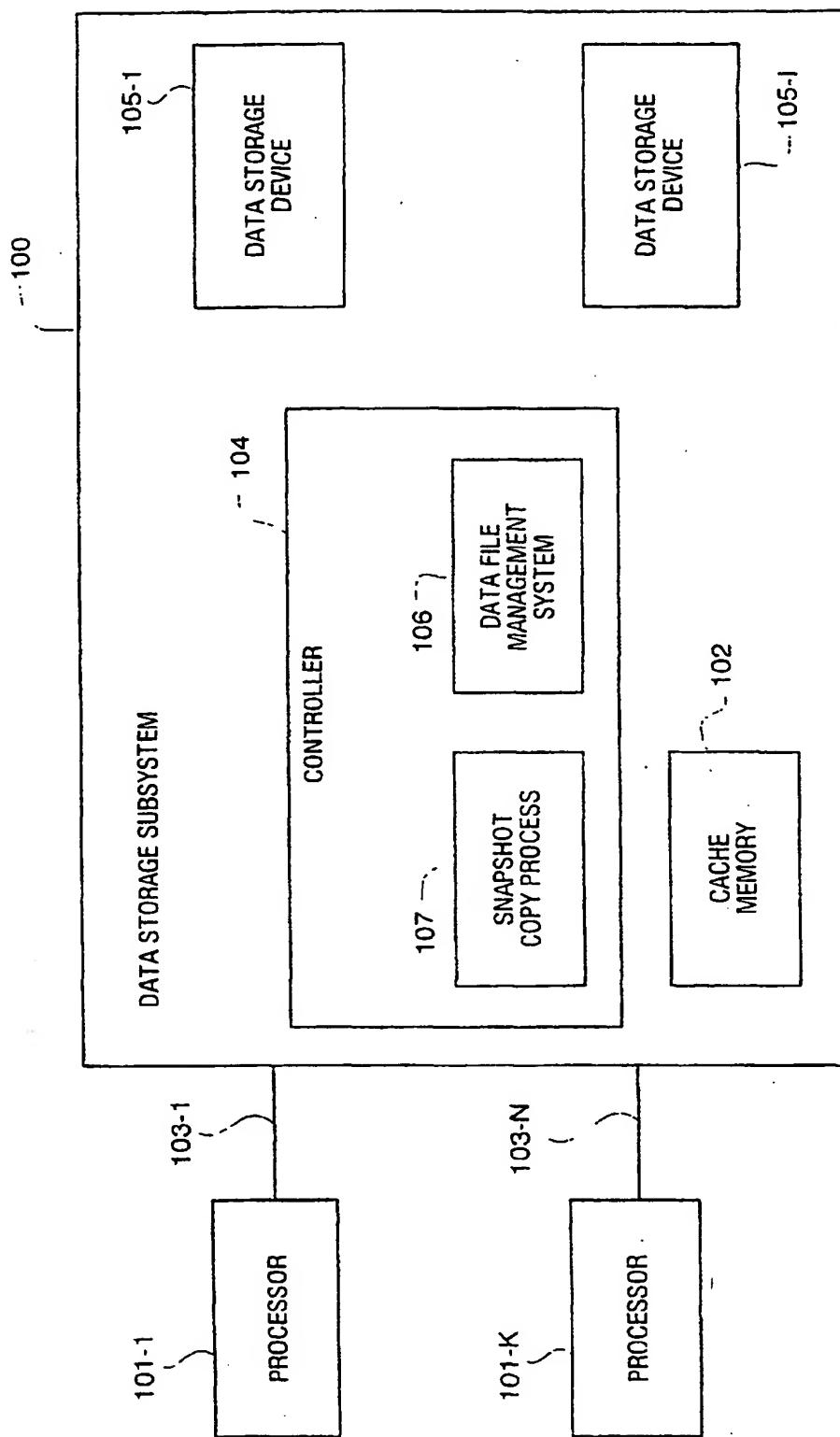


FIG. 1

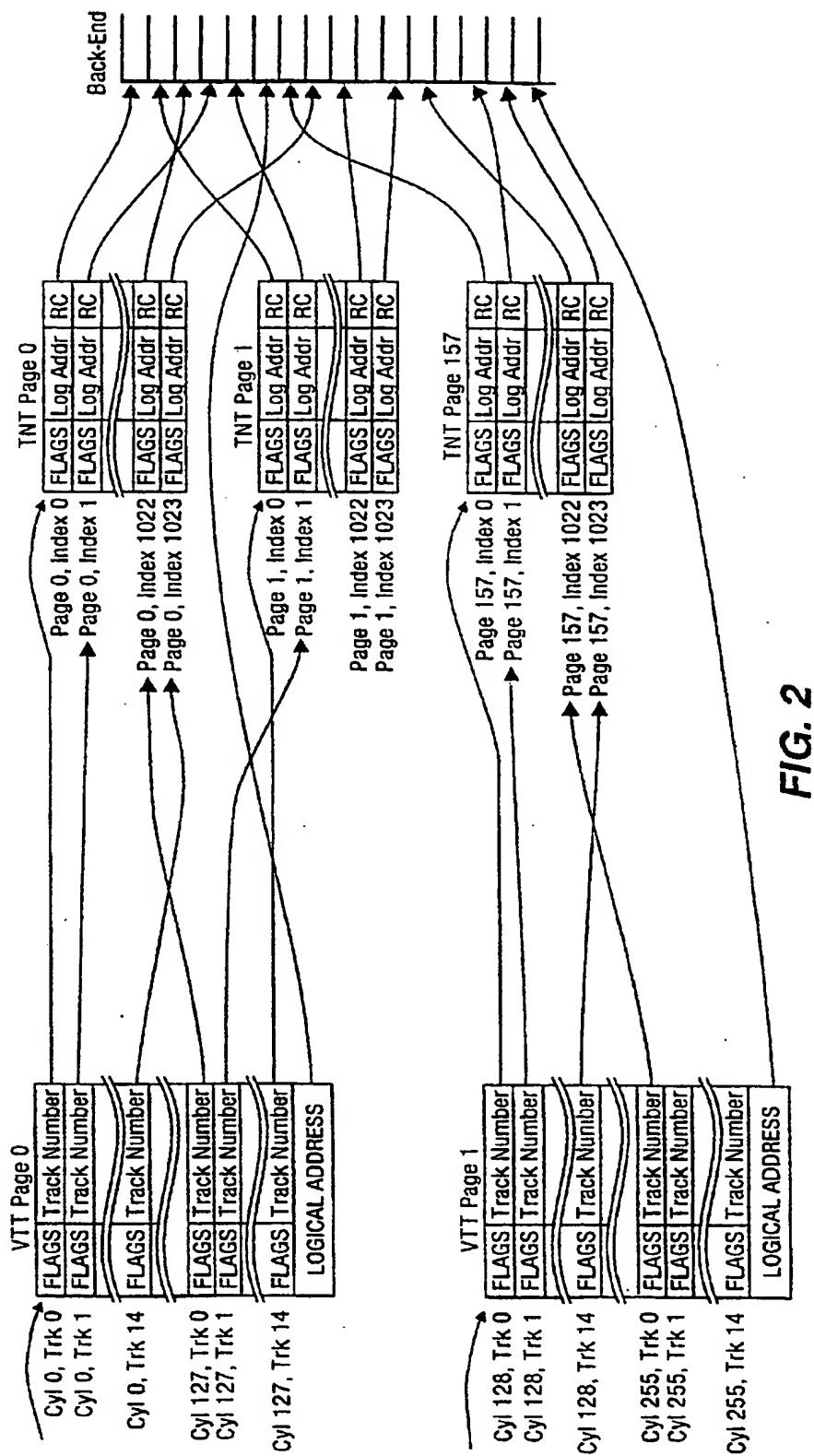
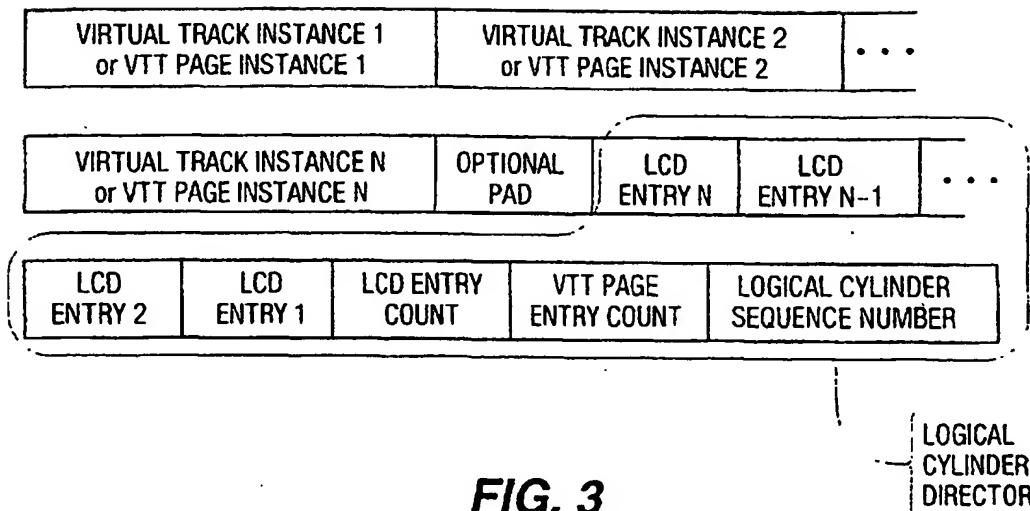


FIG. 2



**FIG. 3**

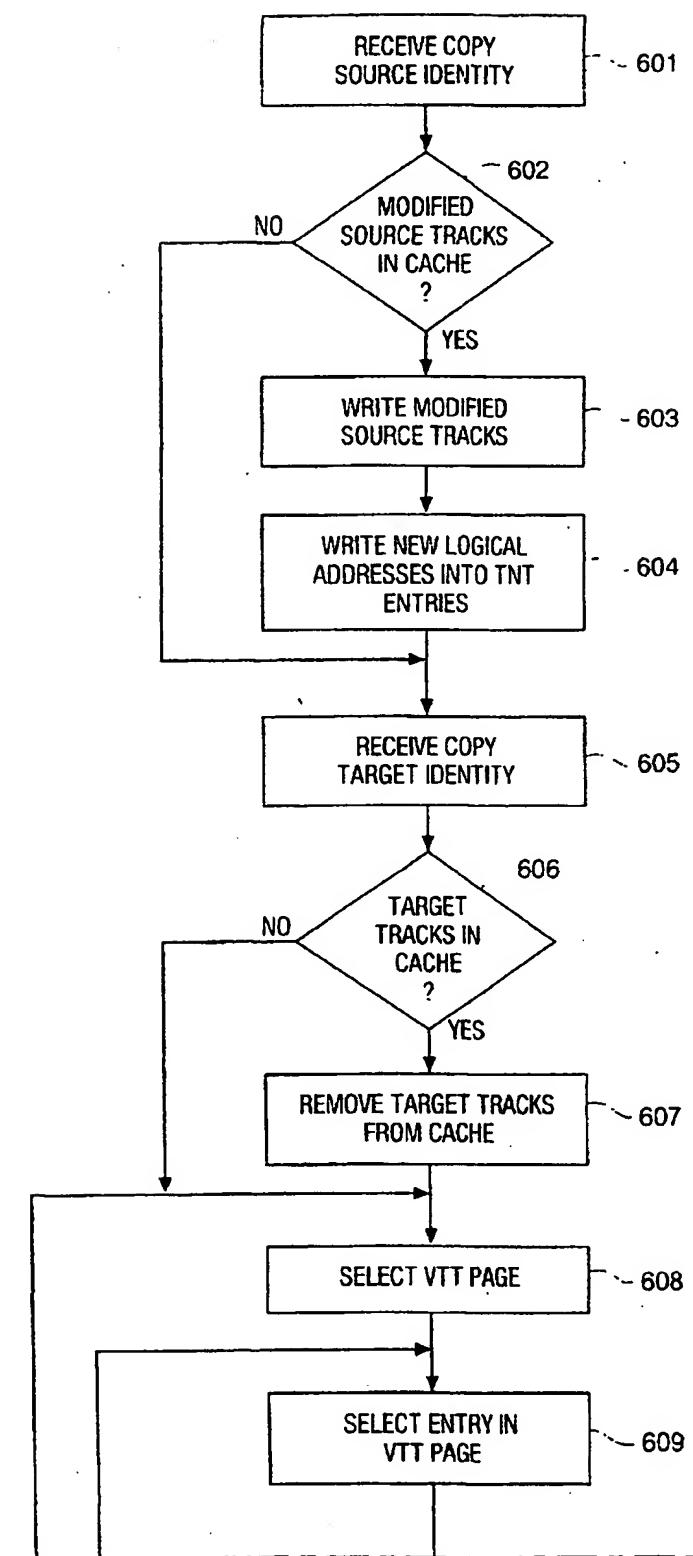
LCD ENTRY TYPE = VIRTUAL TRACK INSTANCE
VIRTUAL TRACK ADDRESS
TRACK NUMBER
LOGICAL ADDRESS
VIRTUAL TRACK INSTANCE LENGTH

**FIG. 4**

LCD ENTRY TYPE = VTT PAGE INSTANCE
VTT PAGE IDENTIFIER
ORIGINAL LCSN
LOGICAL ADDRESS

**FIG. 5**

FIG. 6A



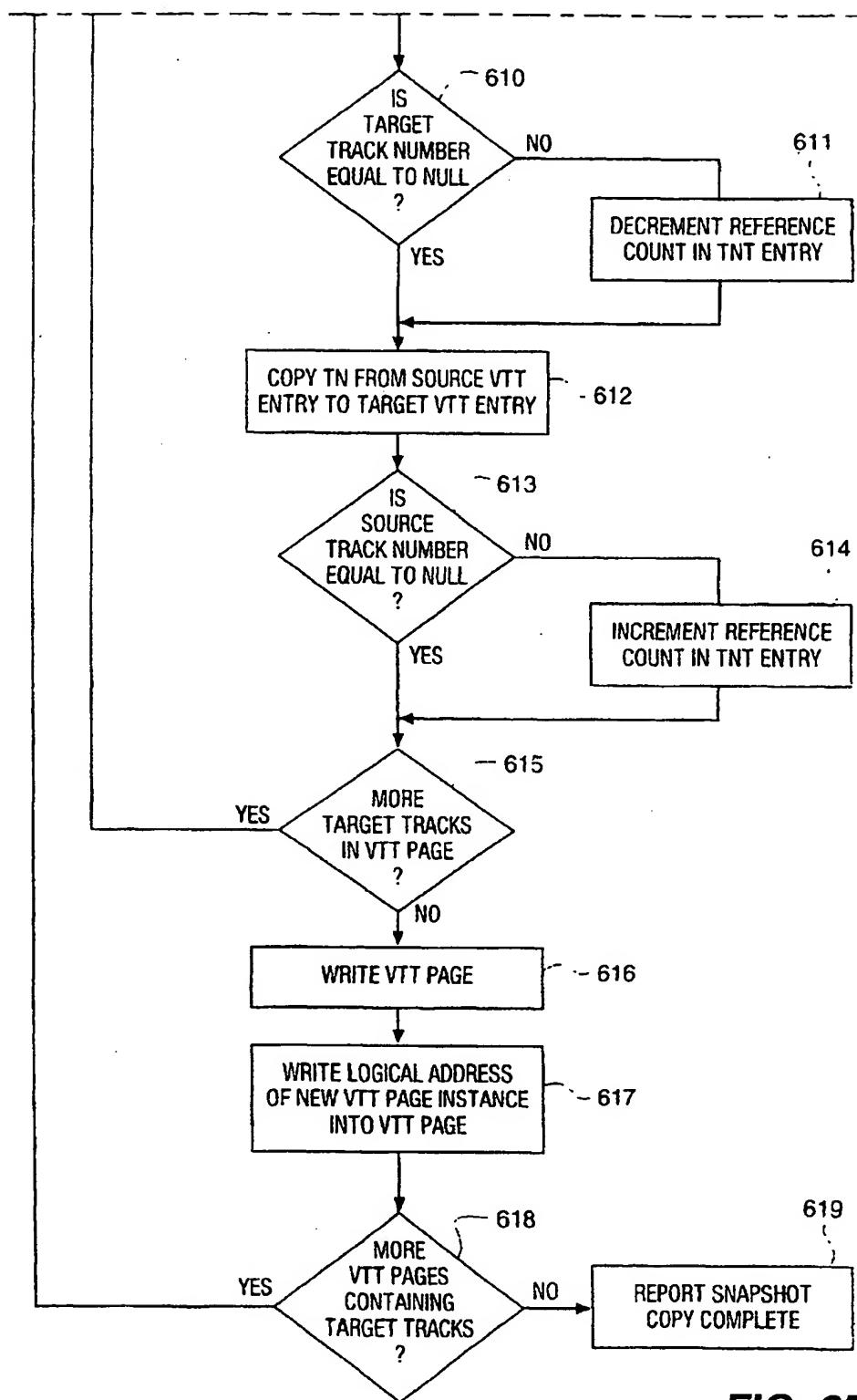


FIG. 6B

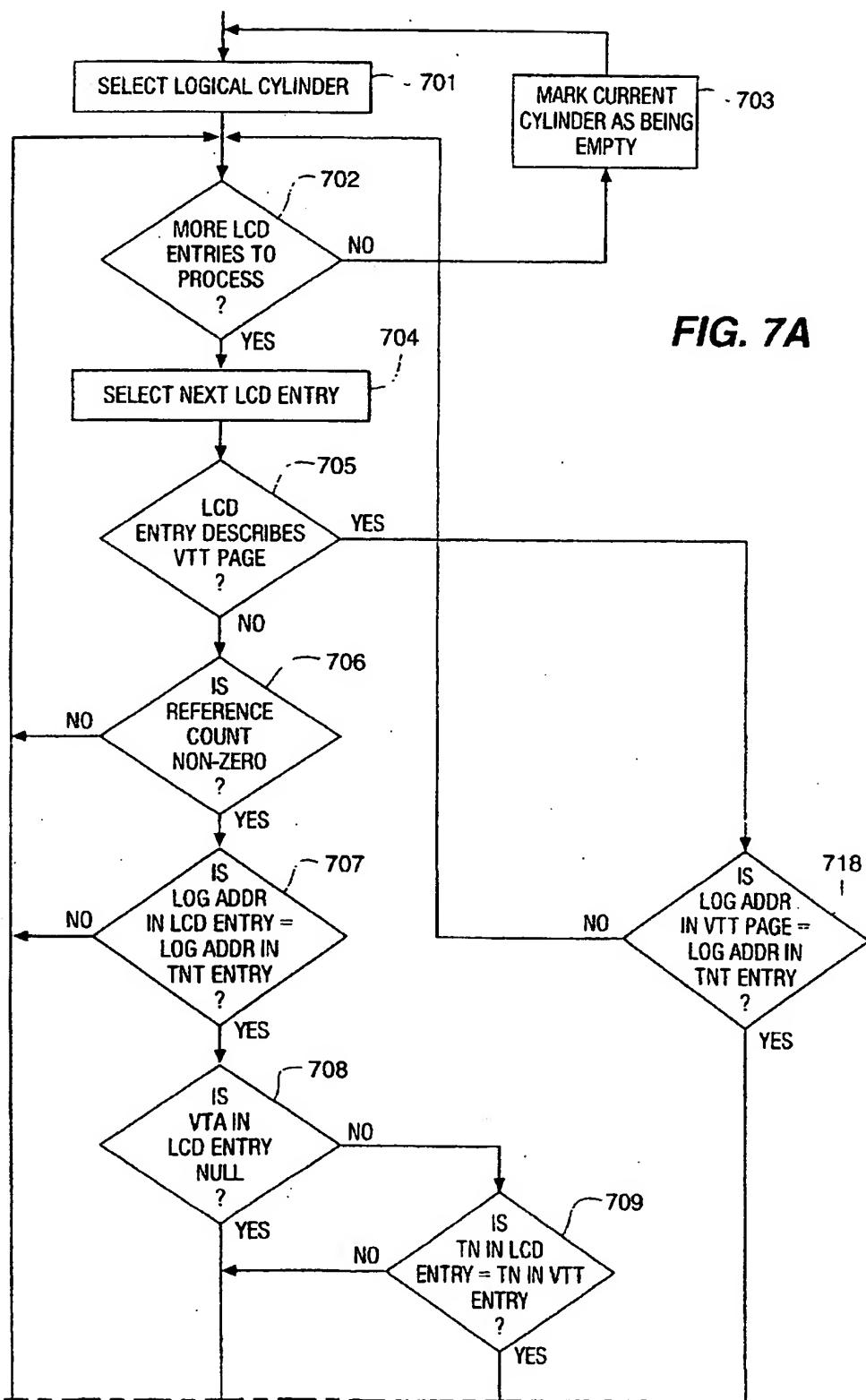


FIG. 7A

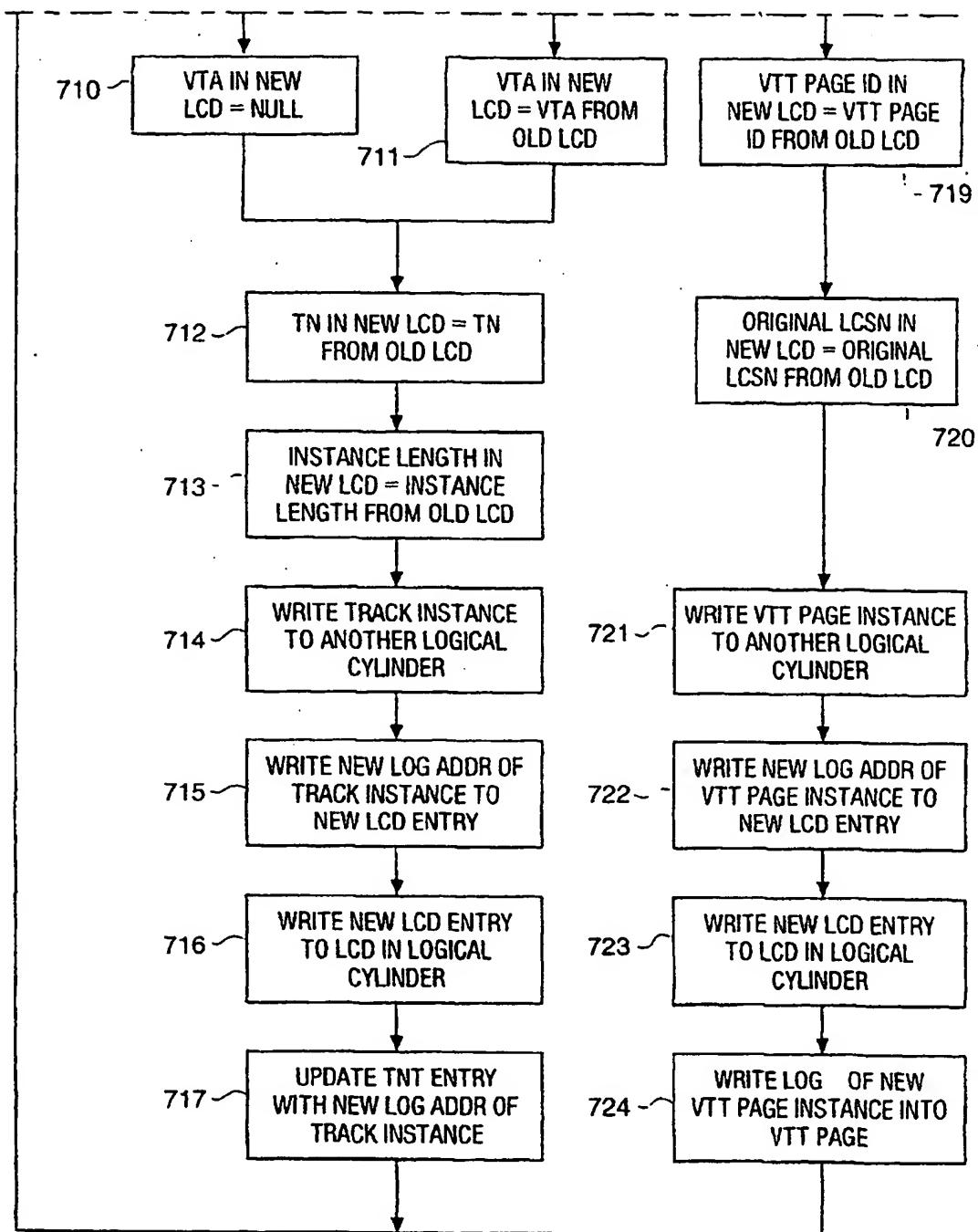


FIG. 7B

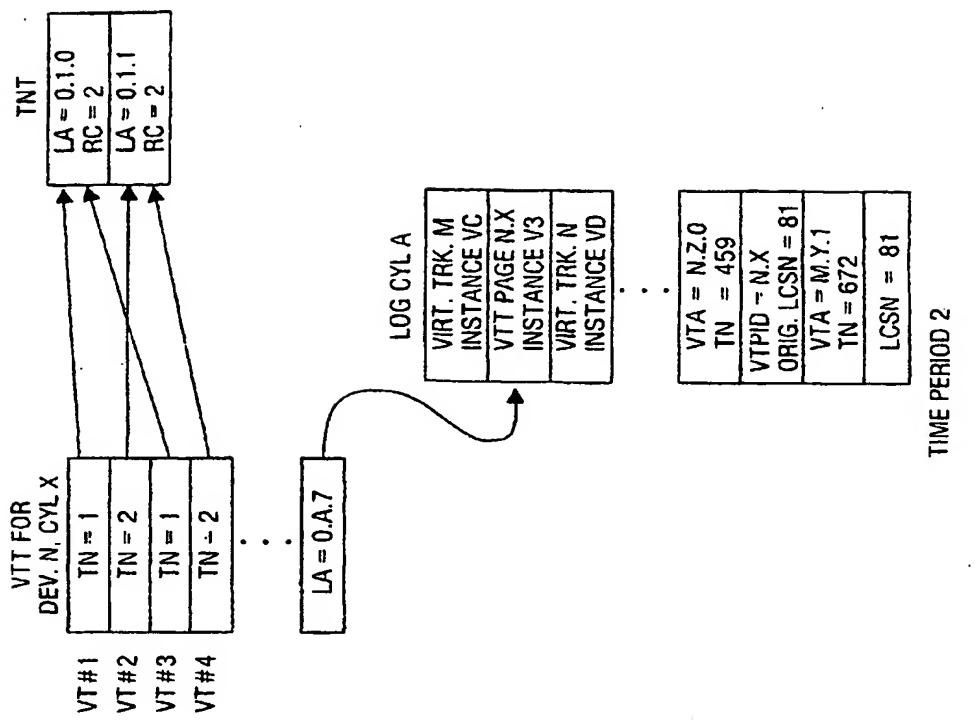


FIG. 9

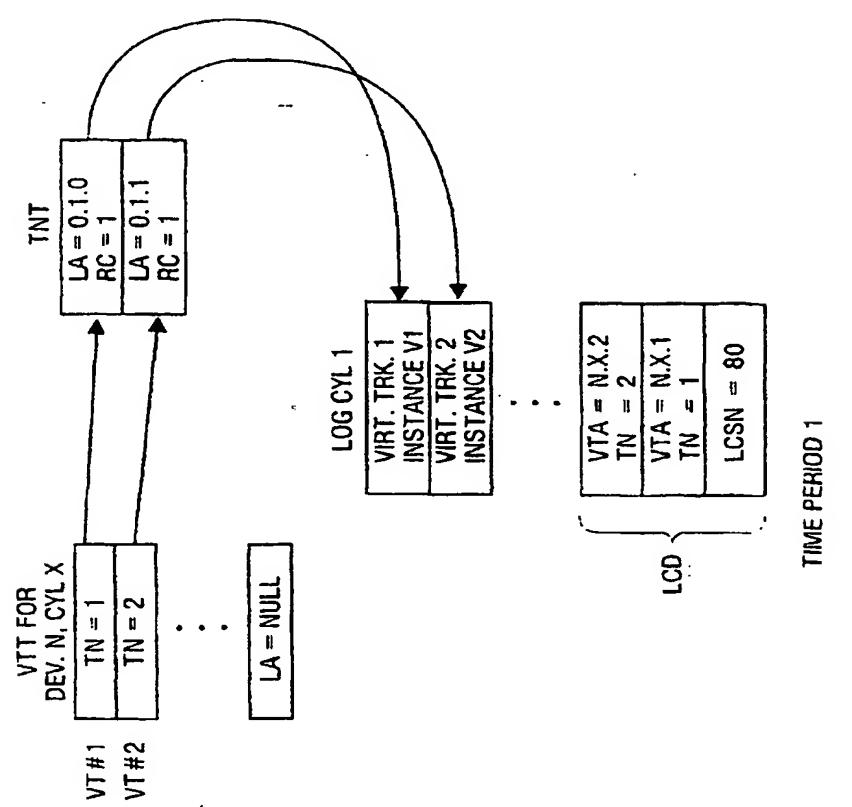
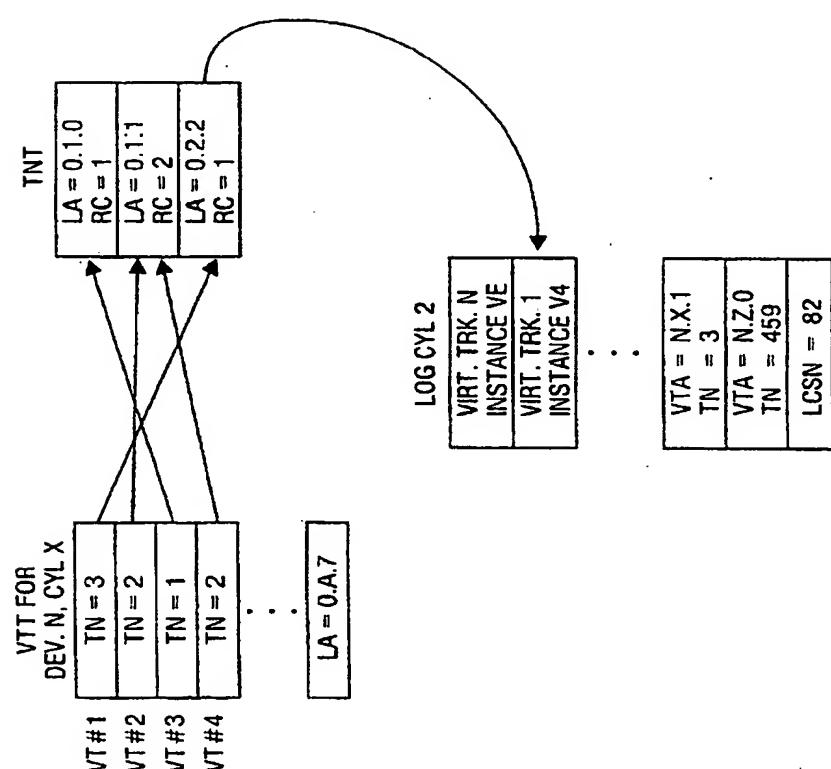
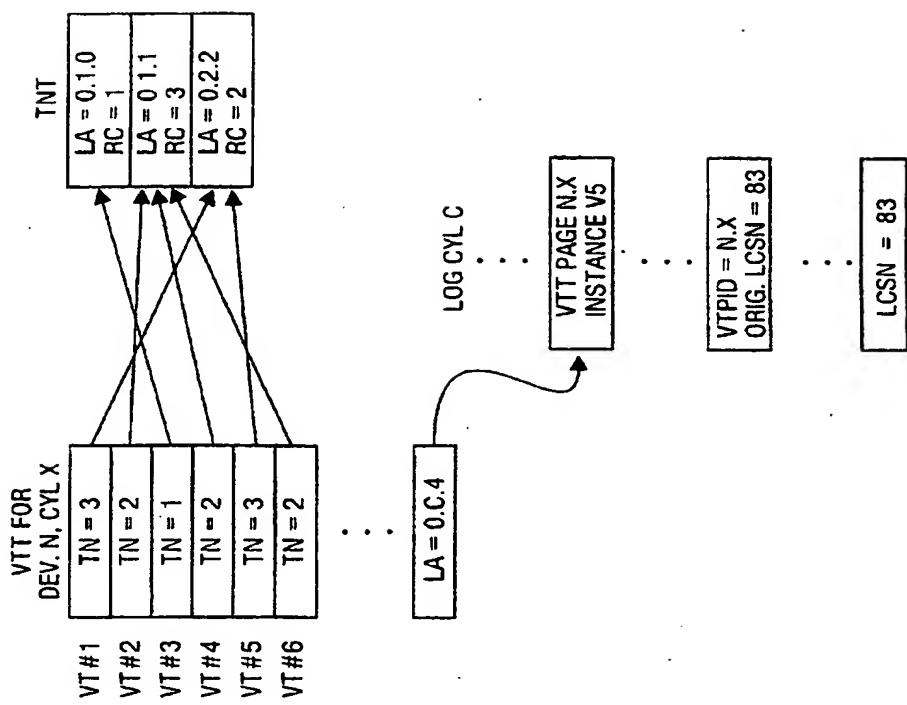


FIG. 8



TIME PERIOD 4

TIME PERIOD 3

FIG. 11

FIG. 10

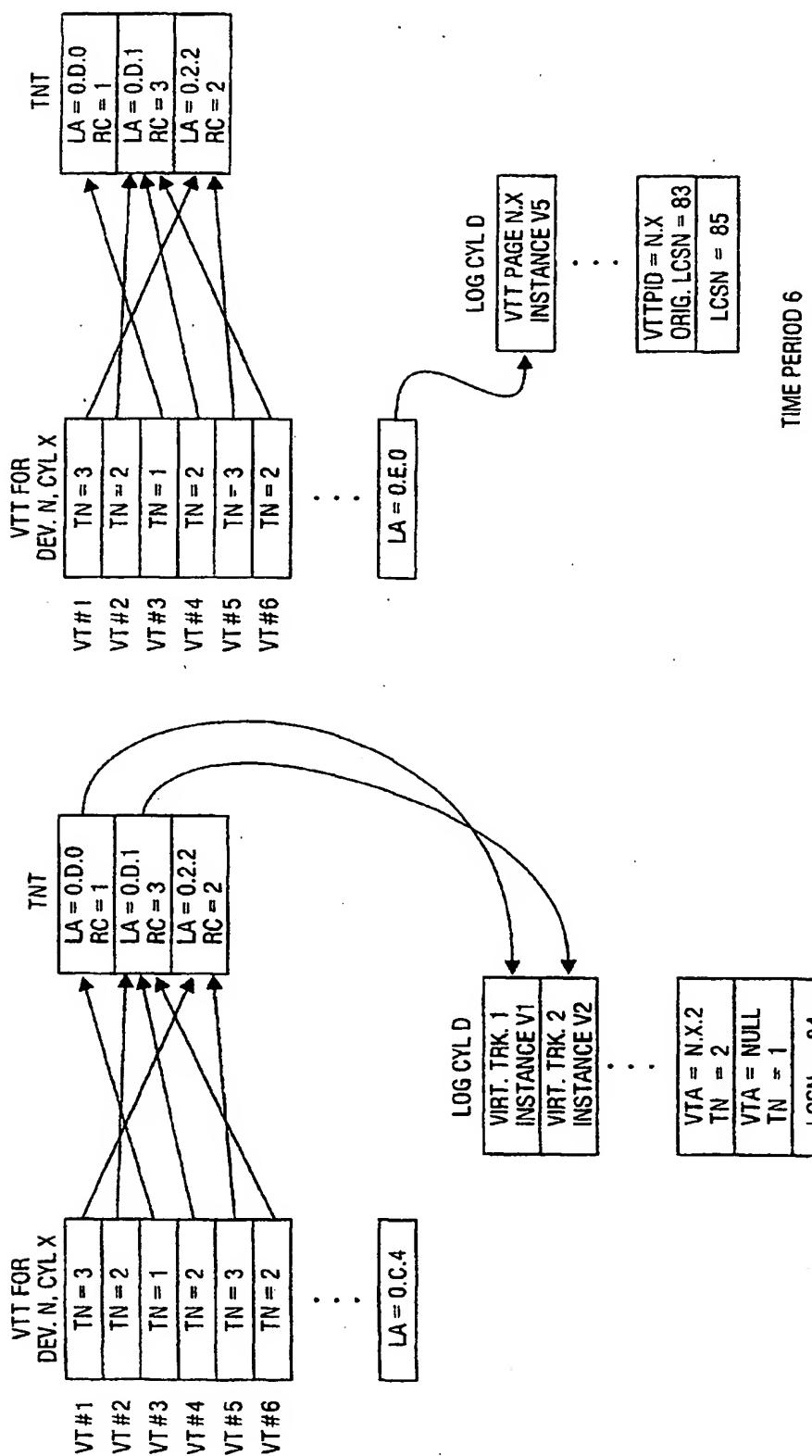


FIG. 12

FIG. 13

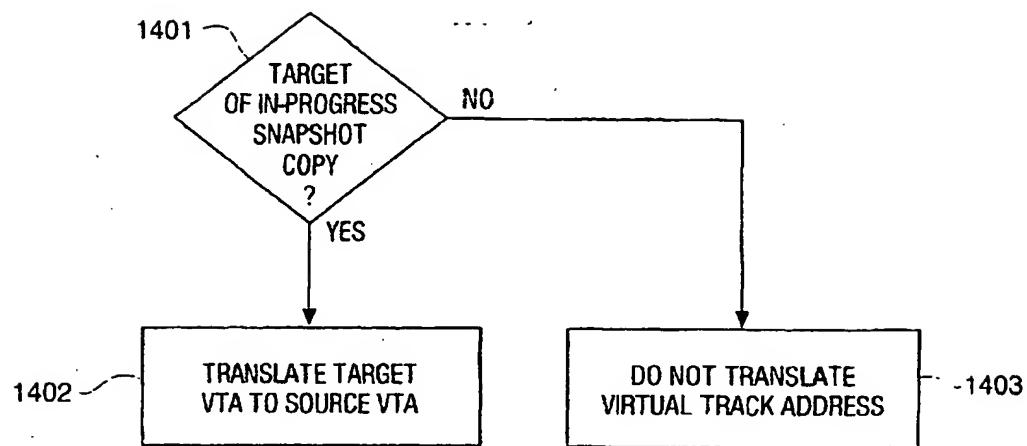


FIG. 14

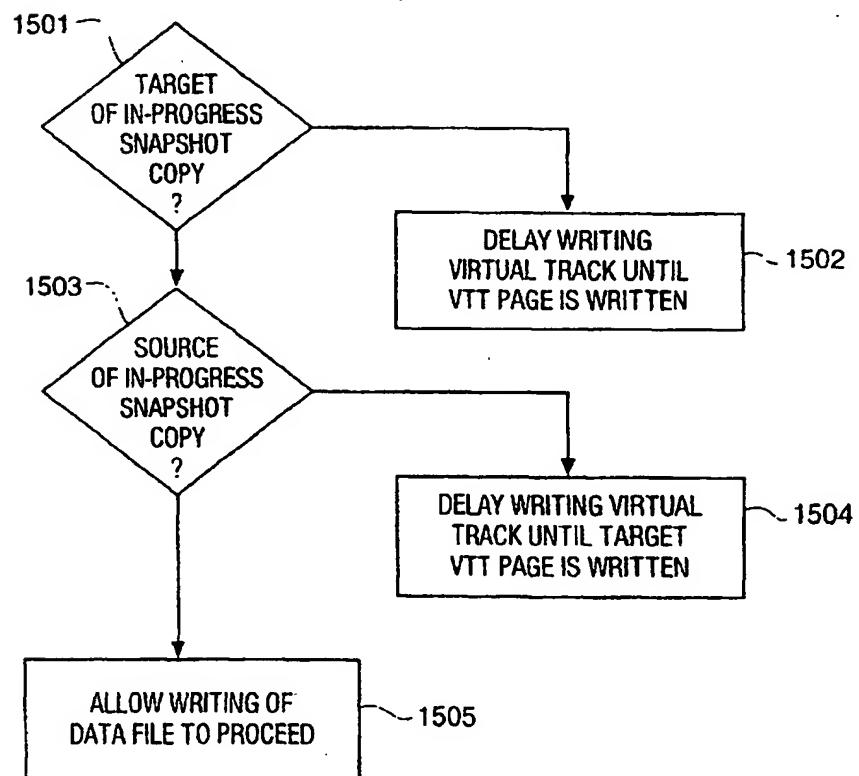
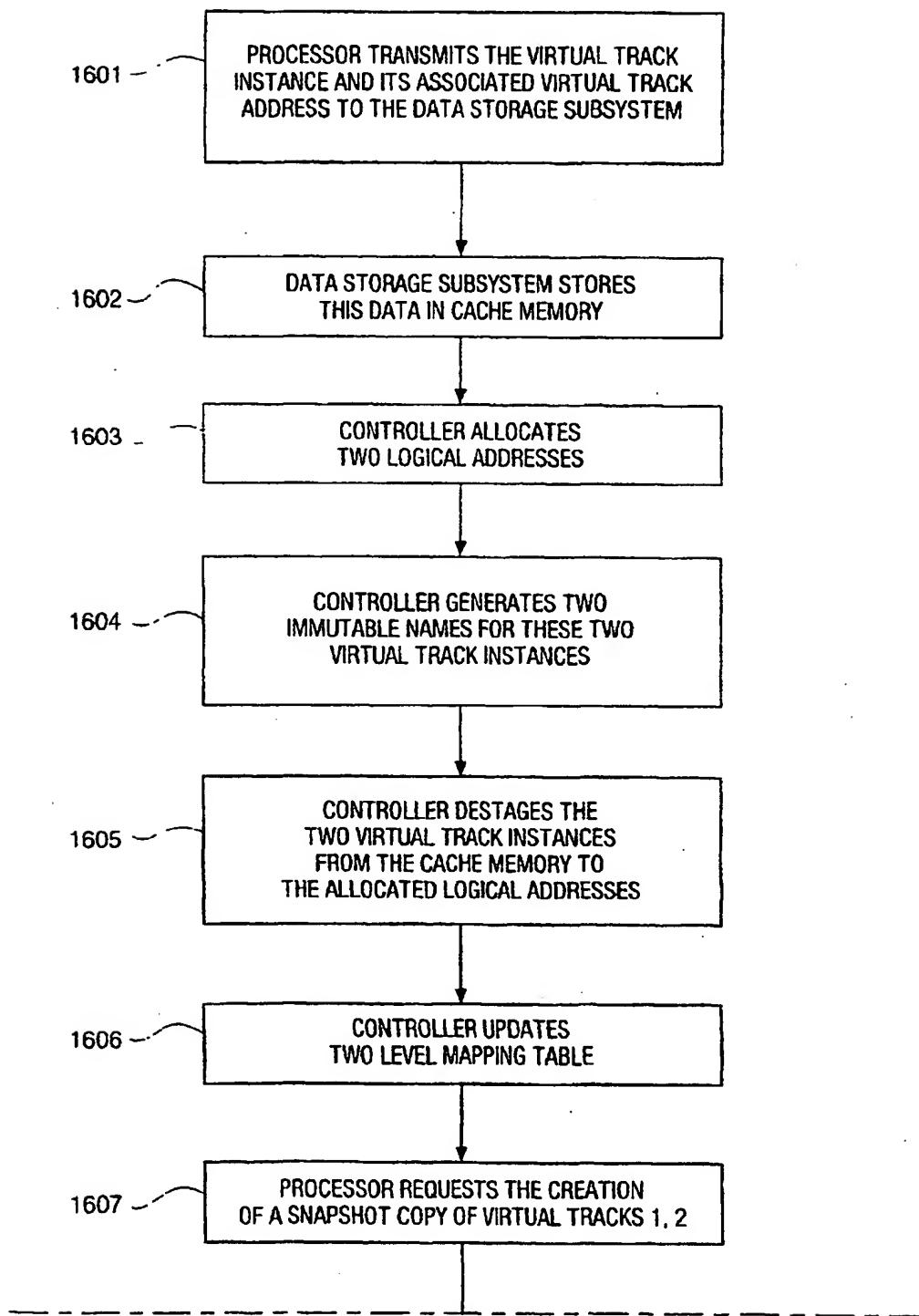


FIG. 15

FIG. 16A



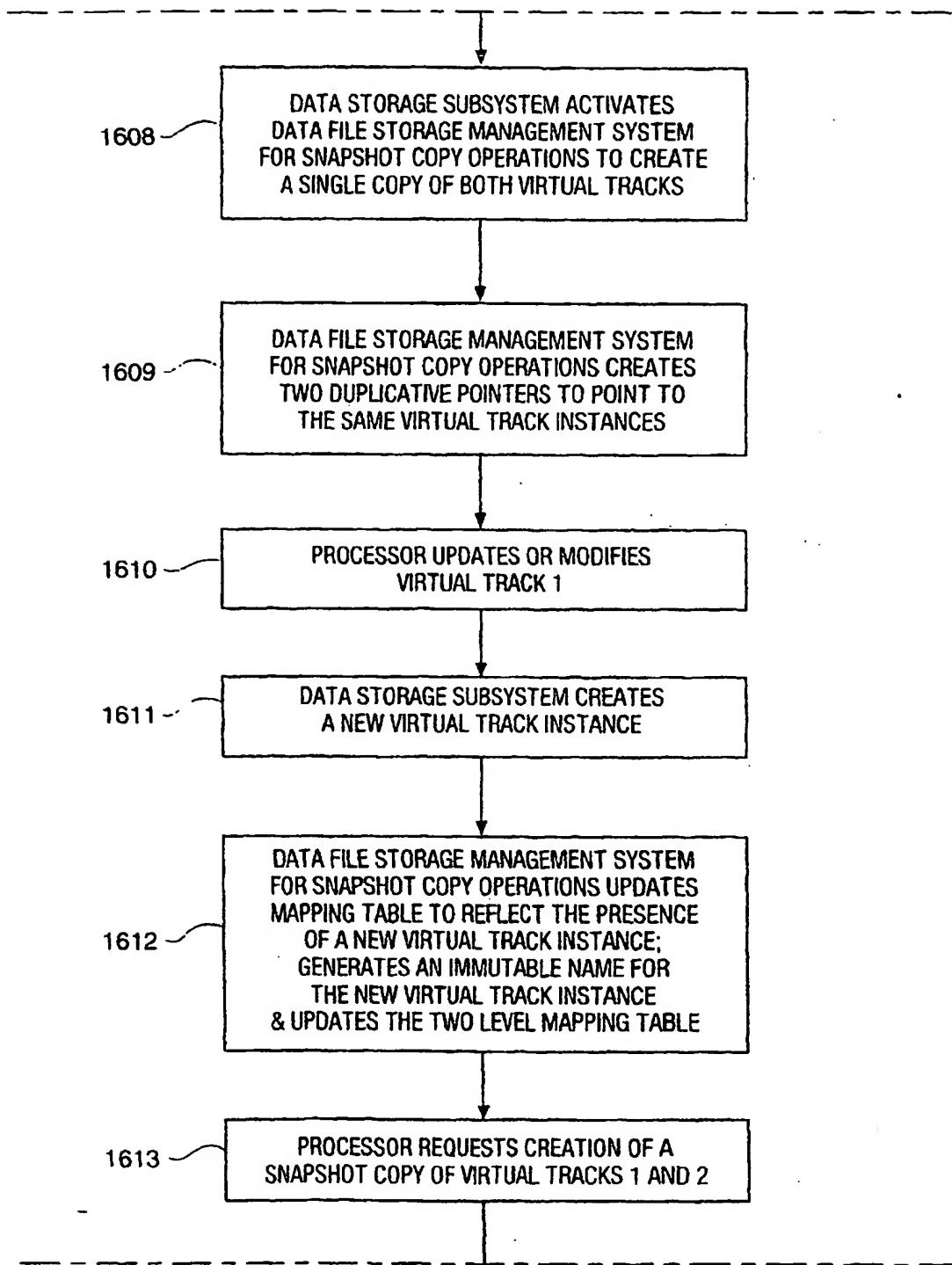
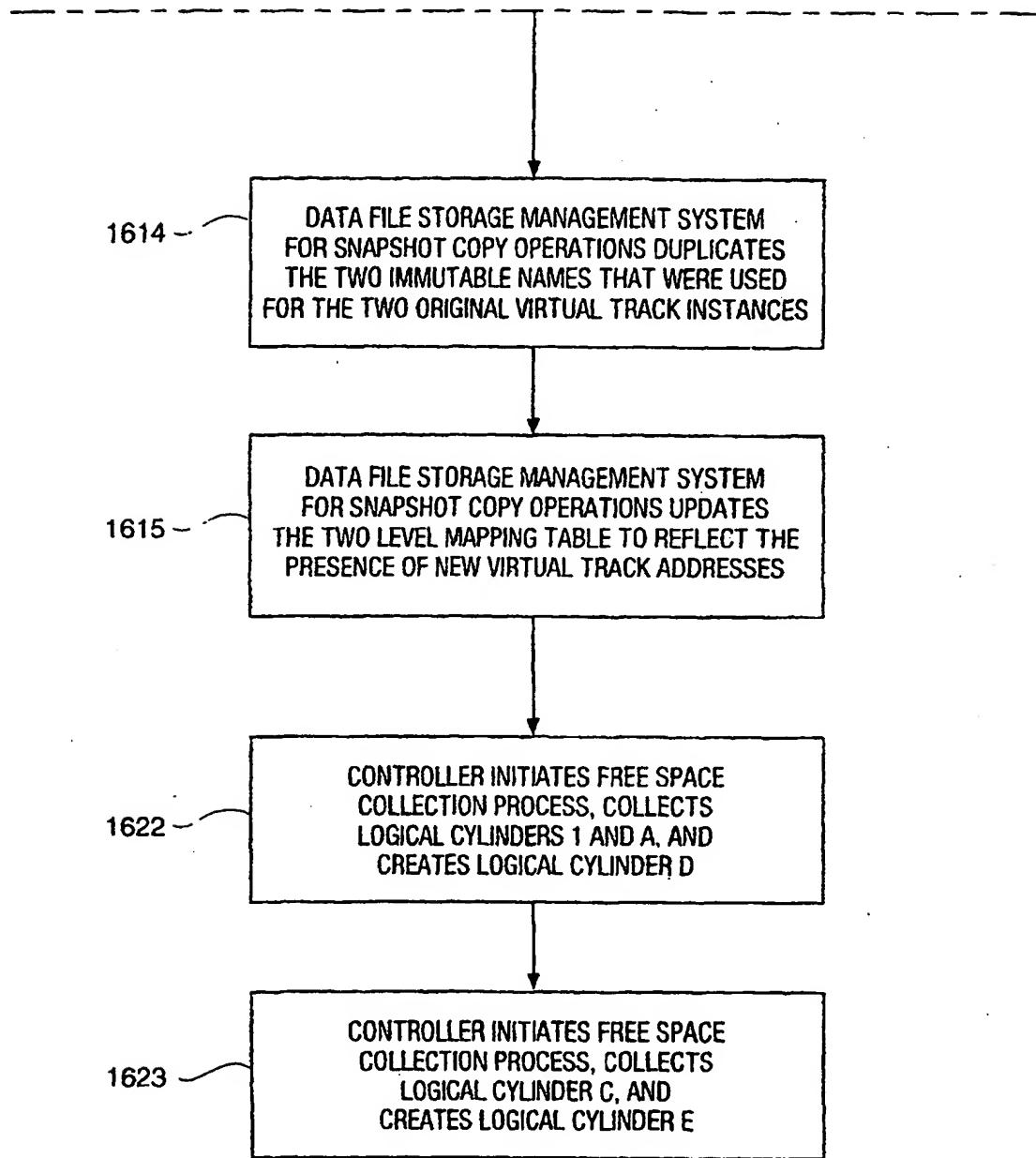


FIG. 16B



**FIG. 16C**